



**FACULTAD 4**

# **Servicio web para la gestión de estaciones de carga de vehículos eléctricos**

Trabajo de diploma para optar por el título de  
Ingeniero en Ciencias Informáticas

**Autor:** Claudia Collazo Gumá

**Tutor:** Ing. Andy Suárez Oña

La Habana, 2023

**DECLARACIÓN DE AUTORÍA**

El autor del trabajo de diploma con título “Servicio web para la gestión de estaciones de carga de vehículos eléctricos”, conceden a la UCI los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declara como único autor de su contenido.

Para que así conste, firma la presente a los 24 días del mes de noviembre del año 2023.

**Claudia Collazo Gumá**



\_\_\_\_\_  
Firma del Autor

**Ing. Andy Suárez Oña.**



\_\_\_\_\_  
Firma del Tutor

## DATOS DE CONTACTO

**Ing. Andy Suárez Oña:** Se graduado el 2019 en Ingeniería en Ciencia Informáticas. Tiene 4 de experiencia desarrollo de software industrial, actualmente se desempeña como jefe de proyecto de la “Plataforma de Servicios de electro movilidad para el ecosistema de carga público de Cuba “. Además de ocupar el cargo de subdirector de Centro Tecnología Interactivas.

Correo electrónico: [aona@uci.cu](mailto:aona@uci.cu);

## RESUMEN

Una de las principales medidas adoptadas por el gobierno cubano ha sido la implementación de programas de electrificación del transporte. No obstante, aún existen retos importantes que deben ser abordados para consolidar la transición hacia un transporte más sostenible en Cuba. Estos desafíos incluyen las congestiones en las estaciones de carga y la deficiencia de la red eléctrica debido a la falta de control sobre la cantidad de energía servida en los puntos de carga. Además, existe una carencia en la gestión de control de acceso donde la seguridad y la administración de recursos es importante. Por ello, en el presente trabajo de diploma se traza como objetivo principal el desarrollo de un servicio web de gestión de cargadores eléctricos que contribuya al control sobre la cantidad de energía administrada en los puntos de carga. Para guiar el desarrollo de la propuesta de solución, se empleó como metodología de desarrollo de software el Proceso Unificado Ágil en su variante UCI. Se realizó una investigación de los sistemas homólogos existentes a nivel internacional. Se utilizó java como lenguaje de desarrollo y PostgreSQL se seleccionó como herramienta de base de datos. Se realizaron una serie de pruebas para garantizar el correcto funcionamiento de las funcionalidades propuestas, asegurando que el sistema cumpla con los requerimientos del cliente. Como resultado se obtuvo un servicio web capaz de gestionar los puntos de carga.

**Palabras clave:** protocolo OCPP, punto de carga, servicio web

**ABSTRACT**

*One of the main measures adopted by the Cuban government has been the implementation of transport electrification programs. However, there are still significant challenges that need to be addressed to consolidate the transition to more sustainable transportation in Cuba. These challenges include congestion at charging stations and the deficiency of the electric grid due to the lack of control over the amount of energy served at charging points. In addition, there is a lack of access control management where security and resource management is important. Therefore, the main objective of the present work is to develop a web service for the management of electric chargers that contributes to the control over the amount of energy managed at the charging points. To guide the development of the proposed solution, the Agile Unified Process in its UCI variant was used as software development methodology. An investigation of the existing international counterpart systems was carried out. Java was used as the development language and PostgreSQL was selected as the database tool. A series of tests were carried out to guarantee the correct functioning of the proposed functionalities, ensuring that the system complies with the client's requirements. As a result, a web service capable of managing the load points was obtained.*

**Key words:** *OCPP protocol, load point, charge point, web service.*

**ÍNDICE DE CONTENIDOS**

INTRODUCCIÓN .....1

CAPÍTULO I: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS DE LOS SERVICIOS WEB PARA LA GESTIÓN DE ESTACIONES DE CARGA DE VEHÍCULOS ELÉCTRICOS .....4

    1.1 Estado del arte..... 4

    1.2 Estudio de sistemas para la gestión de puntos de carga internacionales. .... 10

    1.2 Análisis comparativo de los sistemas de gestión. .... 12

    1.3 Metodología de desarrollo de software ..... 12

        1.3.1 Metodología de desarrollo de software variación de AUP para la UCI ..... 12

    1.4 Lenguajes y herramienta para el modelado de la solución ..... 14

        1.4.1 Lenguaje Unificado de Modelado (UML) v2 ..... 14

        1.4.2 Herramienta para el modelado de la solución ..... 14

        1.4.3 Visual Paradigm v8.0..... 14

    1.5 Tecnologías de implementación ..... 14

        1.5.1 Lenguajes de programación ..... 15

        1.5.2 Marco de trabajo..... 16

        1.5.3 Entorno de desarrollo integrado..... 16

        1.5.4 Sistema Gestor de Base de datos ..... 19

    Conclusiones del capítulo..... 19

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN. ....21

    2.1 Descripción de la propuesta de solución ..... 21

        2.1.1 Modelo conceptual ..... 21

    2.2 Requisitos de software ..... 22

        2.2.1 Requisitos funcionales ..... 22

        2.2.2 Requisitos no funcionales ..... 26

    2.3 Historias de usuario..... 27

        Requisito: Autenticar usuario..... 27

        Requisito: Registrar usuario. .... 28

        Requisito: Eliminar usuario. .... 28

        Requisito: Actualizar usuario. .... 29

        Requisito: Obtener usuario. .... 29

        Requisito: Obtener lista de usuarios. .... 30

        Requisito: Registrar un punto de carga..... 30

Requisito: Actualizar un punto de carga. ....	31
Requisito: Eliminar un punto de carga. ....	31
Requisito: Obtener un punto de carga. ....	32
Requisito: Obtener lista de puntos de carga. ....	32
Requisito: Registrar una sesión de carga. ....	33
Requisito: Actualizar una sesión de carga. ....	33
Requisito: Eliminar una sesión de carga. ....	34
Requisito: Obtener una sesión de carga. ....	34
Requisito: Obtener lista de sesiones de carga. ....	35
Requisito: Registrar un conector de un punto de carga. ....	35
Requisito: Actualizar un conector de un punto de carga. ....	35
Requisito: Eliminar un conector de un punto de carga. ....	36
Requisito: Obtener un conector de un punto de carga. ....	36
Requisito: Obtener lista de los conectores de un punto de carga. ....	37
Requisito: Iniciar una transacción remotamente de un punto de carga. ....	37
Requisito: Detener una transacción remotamente de un punto de carga. ....	38
Requisito: Desbloquear conector remotamente de un punto de carga. ....	38
Requisito: Obtener la configuración de un punto de carga. ....	39
Requisito: Actualizar la configuración de un punto de carga. ....	39
Requisito: Restablecer un punto de carga a su configuración inicial. ....	40
Requisito: Cambiar disponibilidad de un punto de carga. ....	40
Requisito: Procesar las notificaciones de arranque enviadas desde un punto de carga. ....	41
Requisito: Procesar el “ <i>Heartbeat</i> ” enviada desde un punto de carga. ....	41
Requisito: Registrar token. ....	42
Requisito: Eliminar token. ....	42
Requisito: Actualizar token. ....	43
Requisito: Obtener token. ....	43
Requisito: Obtener lista de tokens. ....	44
Requisito: Autorizar un token que solicita cargar en un punto de carga. ....	44
Requisito: Denegar un token que solicita cargar en un punto de carga. ....	45
Requisito: Generar log de todas las operaciones que haga el sistema. ....	45
2.4 Arquitectura del software .....	45
2.4.2 Arquitectura en capas .....	46

2.5 Diagrama de clases.....	48
2.6 Patrones de diseño.....	51
2.6.1 Patrones GRASP (General Responsibility Assignment Software Patterns).....	51
2.6.2 PatronesGoF (Gang of Four).....	54
2.7 Modelo de datos.....	54
2.8 Diagrama de despliegue.....	55
Conclusiones del capítulo.....	56
CAPÍTULO III: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA .....	58
3.1 Pruebas de software.....	58
3.2 Estrategia de pruebas .....	58
3.2.1 Nivel unidad. Métodos de Caja Blanca .....	58
3.2.2 Nivel de integración. Métodos de Caja Negra.....	64
Conclusiones del capítulo.....	67
CONCLUSIONES FINALES .....	68
RECOMENDACIONES.....	69
REFERENCIAS BIBLIOGRÁFICAS.....	70



**ÍNDICE DE TABLAS**

Tabla 1 Perfiles de OCPP (Fuente: (Open Charge Alliance, 2017)) ..... 6

Tabla 2 requisitos funcionales (fuente: elaboración propia)..... 26

Tabla 3 historia de usuario Autenticar usuario (fuente: elaboración propia) ..... 27

Tabla 4 historia de usuario Registrar usuario (fuente: elaboración propia) ..... 28

Tabla 5 historia de usuario Eliminar usuario (fuente: elaboración propia) ..... 28

Tabla 6 historia de usuario Actualizar usuario (fuente: elaboración propia)..... 29

Tabla 7 historia de usuario Obtener usuario (fuente: elaboración propia) ..... 29

Tabla 8 historia de usuario Obtener lista de usuarios (fuente: elaboración propia)..... 30

Tabla 9 historia de usuario Registrar un punto de carga (fuente: elaboración propia) ..... 30

Tabla 10 historia de usuario Actualizar un punto de carga (fuente: elaboración propia) ..... 31

Tabla 11 historia de usuario Eliminar un punto de carga (fuente: elaboración propia)..... 31

Tabla 12 historia de usuario Obtener un punto de carga (fuente: elaboración propia)..... 32

Tabla 13 historia de usuario Obtener lista de puntos de carga (fuente: elaboración propia)..... 32

Tabla 14 historia de usuario Registrar una sesión de carga (fuente: elaboración propia)..... 33

Tabla 15 historia de usuario Actualizar una sesión de carga (fuente: elaboración propia)..... 33

Tabla 16 historia de usuario Eliminar una sesión de carga (fuente: elaboración propia) ..... 34

Tabla 17 historia de usuario Obtener una sesión de carga (fuente: elaboración propia) ..... 34

Tabla 18 historia de usuario Obtener lista de sesiones de carga (fuente: elaboración propia) ..... 34

Tabla 19 historia de usuario Registrar un conector de un punto de carga (fuente: elaboración propia)  
..... 35

Tabla 20 historia de usuario Actualizar un conector de un punto de carga (fuente: elaboración propia)  
..... 35

Tabla 21 historia de usuario Eliminar un conector de un punto de carga (fuente: elaboración propia) 36

Tabla 22 historia de usuario Obtener un conector de un punto de carga (fuente: elaboración propia) 36

Tabla 23 historia de usuario Obtener lista de los conectores de un punto de carga (fuente: elaboración propia) ..... 37

Tabla 24 historia de usuario Iniciar una transacción remotamente de un punto de carga (fuente: elaboración propia) ..... 37

Tabla 25 historia de usuario Detener una transacción remotamente de un punto de carga (fuente: elaboración propia) ..... 38

Tabla 26 historia de usuario Desbloquear conector remotamente de un punto de carga (fuente: elaboración propia) ..... 38

Tabla 27 historia de usuario Obtener la configuración de un punto de carga (fuente: elaboración propia)	39
Tabla 28 historia de usuario Actualizar la configuración de un punto de carga (fuente: elaboración propia)	39
Tabla 29 historia de usuario Restablecer un punto de carga a su configuración inicial (fuente: elaboración propia)	40
Tabla 30 historia de usuario Cambiar disponibilidad de un punto de carga (fuente: elaboración propia)	40
Tabla 31 historia de usuario Procesar las notificaciones de arranque enviadas desde un punto de carga (fuente: elaboración propia)	41
Tabla 32 historia de usuario Procesar el “Heartbeat” enviada desde un punto de carga (fuente: elaboración propia)	41
Tabla 33 historia de usuario Registrar token (fuente: elaboración propia)	42
Tabla 34 historia de usuario Eliminar token (fuente: elaboración propia)	42
Tabla 35 historia de usuario Actualizar token (fuente: elaboración propia)	43
Tabla 36 historia de usuario Obtener token (fuente: elaboración propia)	43
Tabla 37 historia de usuario Obtener lista de tokens (fuente: elaboración propia)	44
Tabla 38 historia de usuario Autorizar un token que solicita cargar en un punto de carga (fuente: elaboración propia)	44
Tabla 39 historia de usuario Denegar un token que solicita cargar en un punto de carga (fuente: elaboración propia)	44
Tabla 40 historia de usuario Generar log de todas las operaciones que haga el sistema (fuente: elaboración propia)	45
Tabla 41 Caso de Prueba de caja blanca para el camino básico 1 (Fuente: Elaboración propia)	61
Tabla 42 Caso de Prueba de caja blanca para el camino básico 2 (Fuente: Elaboración propia)	61
Tabla 43 Caso de Prueba de caja blanca para el camino básico 3 (Fuente: Elaboración propia)	61
Tabla 44 Caso de Prueba de caja blanca para el camino básico 4 (Fuente: Elaboración propia)	61

## ÍNDICE DE FIGURAS

Figura 1 Modelo conceptual de la propuesta de solución (fuente: elaboración propia).....	22
Figura 2 Diagrama de arquitectura por capas del sistema (fuente: elaboración propia). ....	47
Figura 3 Diagrama de clases Gestionar token (fuente: elaboración propia). ....	48
Figura 4 Diagrama de clases Gestionar configuración (fuente: elaboración propia). ....	49
Figura 5 Diagrama de clases Gestionar sesiones de carga (fuente: elaboración propia). ....	49
Figura 6 Diagrama de clases correspondiente a la interacción entre el sistema y el punto de carga (fuente: elaboración propia). ....	50
Figura 7 Diagrama de clases Gestionar conectores (fuente: elaboración propia).....	51
Figura 8 Patrón controlador (fuente: elaboración propia). ....	52
Figura 9 Patrón bajo acoplamiento (fuente: elaboración propia). ....	52
Figura 10 Patrón alta cohesión (fuente: elaboración propia). ....	53
Figura 11 Patrón experto (fuente: elaboración propia) .....	54
Figura 12 Modelo de datos (fuente: elaboración propia) .....	55
Figura 13 Diagrama de despliegue (fuente: elaboración propia) .....	56
Figura 14 Implementación del método "sendUnlockConnectorRequest" con los flujos identificados (Fuente: elaboración propia) .....	59
Figura 15 Grafo de flujo asociado a la implementación del método sendUnlockConnectorRequest (Fuente: elaboración propia) .....	60
Figura 16 Implementación del caso de prueba #4. (Fuente: elaboración propia) .....	62
Figura 17 Resultados arrojados por la herramienta Junit 5 (Fuente: elaboración propia) .....	62
Figura 18 Resultados arrojados por la herramienta Junit 5 en la segunda iteración. (Fuente: elaboración propia) .....	63
Figura 19 Resultados de las pruebas unitarias por iteración (Fuente: elaboración propia).....	64
Figura 20 Resultados arrojados por la herramienta Postman (Fuente: elaboración propia) .....	65
Figura 21 Resultado arrojados por la herramienta Postman en la segunda iteración. (Fuente: elaboración propia) .....	66
Figura 22 Resultados de las pruebas de integración por iteración (Fuente: elaboración propia) .....	67

## INTRODUCCIÓN

En la última década, la electromovilidad ha experimentado un notable crecimiento a nivel mundial, impulsada por la creciente conciencia ambiental, los avances tecnológicos y la imperante necesidad de abordar el cambio climático. Numerosos países han implementado medidas para promover la adopción de vehículos eléctricos (VE) como parte central de sus estrategias para reducir las emisiones de gases de efecto invernadero y disminuir la dependencia de combustibles fósiles.

En Cuba, el sector de transporte automotor ha sido históricamente vital para el desarrollo económico y social, no obstante factores como la elevada obsolescencia de los vehículos de combustión, las dificultades para adquirir piezas de repuesto, la alta dependencia de importaciones de combustible y la preocupación por la contaminación ambiental establecen condiciones propicias para impulsar la electromovilidad en la isla.

Según el Ministerio de Ciencia, Tecnología y Medio Ambiente de nuestro país se espera una reducción en el uso de combustibles fósiles en los vehículos terrestres en un 50 % para 2030. Esto tiene como objetivo evitar anualmente la emisión de un millón de toneladas de huella de carbono o su equivalente en dióxido de carbono (CO<sub>2</sub>eq). Para alcanzar esta meta, la dirección del país aprobó en 2019 la proyección de vehículos eléctricos, estableciendo las pautas para la introducción paulatina de estos medios de transporte.

El Ministerio del Transporte, en colaboración con diversas entidades, está avanzando en un proyecto de norma para la infraestructura de carga de vehículos eléctricos con energía fotovoltaica. Esto incluirá la instalación de puntos de carga en lugares estratégicos como terminales de ómnibus, hospitales y centros comerciales (Periódico Granma Por Susana Antón Rodríguez, 2022).

No obstante, en el creciente ecosistema de vehículos eléctricos, la falta de una gestión eficiente de los puntos de carga se presenta como un desafío clave. La optimización ineficiente de la infraestructura de carga puede conducir a varios problemas que afectan tanto a los operadores de estaciones como a los usuarios. Entre las cuestiones más destacadas se encuentran las congestiones en las estaciones de carga aumentando significativamente los tiempos de espera y afectando negativamente la experiencia del usuario, el desperdicio de recursos energéticos y espaciales donde la ausencia de una distribución eficiente de la energía disponible puede resultar en picos de demanda energética, llevando a un uso ineficiente de recursos y aumentando los costos operativos. La ausencia de autenticación permite que cualquier individuo pueda cargar su vehículo sin autorización, lo que podría resultar en el uso no autorizado de los servicios de carga y, en última instancia, en pérdidas económicas para los operadores. La falta de control sobre la cantidad de energía servida podría llevar a situaciones de sobrecarga, poniendo en riesgo la estabilidad de la red eléctrica local y aumentando la probabilidad de

fallas. Sin un sistema de autenticación y control, es difícil rastrear el número de usuarios que están cargando simultáneamente, lo que puede llevar a congestiones no registradas y a una gestión ineficiente de la infraestructura.

A partir de la problemática antes descrita se genera la necesidad de resolver el siguiente **problema de investigación**: ¿Cómo garantizar el control y acceso a una red de cargadores eléctricos?

Del problema anterior se define el siguiente **objeto de estudio**: sistemas de gestión de cargadores eléctricos en el sector de la electromovilidad.

Como **campo de acción**: servicio web para la gestión de cargadores eléctricos mediante el protocolo OCPP para los servicios de electromovilidad.

Con el fin de solucionar el problema planteado se define como **objetivo general**: desarrollar un servicio web para la gestión de estaciones de carga de vehículos eléctricos para el servicio de electromovilidad cubano.

Para dar cumplimiento al objetivo propuesto se han derivado un conjunto de **objetivos específicos**:

- Elaborar el marco teórico de la investigación mediante el estudio de los referentes teóricos sobre el desarrollo de servicios web para la gestión de estaciones de carga de vehículos eléctricos.
- Realizar la identificación de los requisitos, análisis y diseño de la propuesta de solución.
- Validar los resultados de la investigación aplicando pruebas de software.

Definiéndose como idea a defender: Desarrollar un servicio web para la gestión de estaciones de carga de vehículos eléctricos ofrece numerosos beneficios que contribuyen significativamente a la eficiencia y la efectividad de la infraestructura de carga. Los operadores pueden monitorear y controlar remotamente el estado de cada estación, optimizando su rendimiento y resolviendo problemas de manera proactiva. Un servicio web proporciona datos en tiempo real sobre la actividad de las estaciones de carga. Esto incluye la disponibilidad de las estaciones, el estado de carga de los vehículos, y el consumo de energía. La información en tiempo real permite una toma de decisiones más efectiva.

Para obtener los conocimientos necesarios que hagan posible el cumplimiento del objetivo trazado, se lleva a cabo una investigación en las que se utilizan algunos de los métodos científicos existentes, tanto teóricos como empíricos.

#### **Métodos teóricos:**

- Modelación: es empleada en la representación mediante diagramas de las características, procesos y componentes de la solución propuesta, así como la relación existente entre ellos.
- Analítico-sintético: se emplearon en el proceso de análisis documental y revisión bibliográfica, con el objetivo de extraer las ideas esenciales que permitirán fundamentar desde el punto de vista teórico la investigación, así como la propuesta que se realiza.

- Hipotético-deductivo: se utilizó para guiar la investigación desde el planteamiento del problema hasta la verificación de la solución a partir de las validaciones, orientando la secuencia lógica de las tareas que se realizaron.

**Métodos empíricos:**

- Entrevista: se emplea en encuentros con el cliente para conocer la necesidad del desarrollo de la propuesta de solución, definir sus funcionalidades e identificar a la vez particularidades de cada usuario y las restricciones que se imponen.

**Estructura de la investigación por capítulos**

El presente trabajo de investigación se encuentra estructurado en 3 capítulos:

**Capítulo 1: Fundamentos y Referentes Teórico-Methodológicos de los servicios web para la gestión de estaciones de carga de vehículos eléctricos.**

Este capítulo se centra en los aspectos teóricos y conceptos fundamentales relacionados con el objeto de estudio y el campo de acción de la investigación. Se analizan las soluciones existentes tanto en el ámbito internacional como en el nacional y se describe el entorno de desarrollo a partir de la fundamentación del uso de la metodología, tecnologías y herramientas propuestas para el desarrollo de la propuesta de solución.

**Capítulo 2: Análisis y Diseño de la propuesta de solución.**

En este capítulo se llevará a cabo la caracterización de la propuesta de solución. Se especifican los requisitos funcionales y no funcionales a partir de la metodología seleccionada, y se realiza el diseño ingenieril donde se describen los patrones de diseño definidos como buenas prácticas durante el ciclo de desarrollo del software, la realización del modelado de diagramas, los elementos fundamentales del diseño y de la arquitectura que se deben tener en cuenta para llegar propuesta solución.

**Capítulo 3: Validación de la solución propuesta.**

En este capítulo se valida el grado de calidad y fiabilidad de los resultados obtenidos en el desarrollo de la presente investigación. Para la validación de la solución se define una estrategia de prueba aplicando 2 niveles de pruebas.

Finalmente, se presentan las conclusiones, recomendaciones, referencias bibliográficas y anexos derivados de la investigación.

## CAPÍTULO I: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS DE LOS SERVICIOS WEB PARA LA GESTIÓN DE ESTACIONES DE CARGA DE VEHÍCULOS ELÉCTRICOS

Este capítulo está destinado a brindar una panorámica de todo lo referente al servicio web para la gestión de estaciones de carga de vehículos eléctricos, concepto y funciones del mismo. También se describen las herramientas, tecnologías y la metodología que se utilizará para el proceso de desarrollo de software.

### 1.1 Estado del arte

Una estación de carga es un sistema físico donde los vehículos eléctricos pueden cargar su energía. Estas estaciones pueden contar con uno o varios conectores, que se refieren a tomas de corriente eléctrica gestionadas y operadas de forma independiente en un punto de carga (Open Charge Alliance, 2017). Una red de carga de vehículos eléctricos es una infraestructura diseñada para suministrar energía eléctrica a los vehículos eléctricos (VE) para recargar sus baterías. Estas redes están compuestas por estaciones de carga que se ubican en diferentes lugares, como: estacionamientos públicos, estaciones de servicio, centros comerciales o incluso en las calles (*Global EV – Analysis*, 2021). Una red de carga de vehículos eléctricos permitirá a los operadores obtener información en tiempo real sobre el rendimiento de sus estaciones de carga de vehículos eléctricos, gestionar el estado de carga, habilitar herramientas de fijación dinámica de precios, procesar pagos y detectar instantáneamente fallas y emitir tickets para servicio. El software de gestión de red toma la información de la estación de carga y la comunica al sistema central en una empresa de servicios públicos, municipio u otro administrador. Desde el lado del consumidor, esto permite servicios como facturación, control de acceso, autenticación y pago. Para el anfitrión del sitio, permite establecer políticas de precios y uso, así como utilizar datos (greenlots, 2018).

Ante el crecimiento acelerado de la adopción de vehículos eléctricos en todo el mundo, se hizo evidente la importancia de contar con una infraestructura de carga que fuera accesible, interoperable y eficiente por lo que surge la Alianza de Carga Abierta (OCA).

**La Alianza de Carga Abierta** es un consorcio global de líderes de infraestructura de vehículos eléctricos, tanto públicos como privados, que se han unido para promover estándares abiertos, incluyendo el Protocolo de Punto de Carga Abierto (OCPP, por sus siglas ingles) y el Protocolo Abierto de Carga Inteligente (OSCP, por sus siglas ingles). La OCA proporciona protocolos de comunicación abiertos e interoperables para la infraestructura de carga de vehículos eléctricos, con el fin de respaldar la funcionalidad necesaria para los sistemas avanzados de gestión de carga en la actualidad. Su misión

es fomentar el desarrollo, la adopción y el cumplimiento global de los protocolos de comunicación para la carga de vehículos eléctricos y estándares relacionados a través de la colaboración, la educación, las pruebas y la certificación (greenlots, 2018). Se analiza OCPP porque tiene un enfoque directo en la gestión de las estaciones de carga.

**El Protocolo de Punto de Carga Abierto** es un protocolo de comunicación abierto entre puntos de carga (término técnico para estaciones de carga de vehículos eléctricos) y un sistema central que gestiona los puntos de recarga y dispone de la información para autorizar a los usuarios a utilizar sus puntos de carga (Priyasta et al., 2023).

Un protocolo abierto permite que la estación de carga y el sistema central interoperen sin la necesidad de una interfaz o pasarela propietaria. Ambos hablan el mismo idioma y no se requiere traducción. Un protocolo cerrado es propietario y no está abierto a la comunicación con otros productos sin una interfaz o pasarela (van Amstel et al., 2021).

Algunas de las principales ventajas del protocolo OCPP con respecto a los protocolos privados son:

- Soportar al propietario de una estación de carga para cambiar de operador de red cuando lo desee, evitando así que los activos de la estación de carga queden varados.
- Permitir que las comunicaciones comunes entre la estación de carga y el proveedor de servicios de red también se aprovechen para proporcionar servicios de red de manera rentable.
- Fomentar que los clientes sean propietarios de vehículos eléctricos al permitir un acceso uniforme a estaciones de carga, servicios de itinerancia y facturación (van Amstel et al., 2021).

Las funcionalidades de OCPP se pueden utilizar para compartir la identidad de un punto de carga, informar sobre la condición actual de un punto de carga, autorizar a los usuarios de vehículos eléctricos que desean comenzar la carga, informar que ha comenzado una sesión de carga, reportar valores del medidor de una sesión de carga periódicamente y notificar que una sesión de carga ha finalizado (Priyasta et al., 2023).

Esta investigación se centra en la versión OCPP 1.6. Esta versión ha sido probada en el campo, adoptada masivamente por fabricantes y operadores, y es compatible con una amplia variedad de equipos existentes. Su adopción generalizada y la evolución gradual a lo largo del tiempo han contribuido a su continua relevancia (Lesjak, 2023).

OCPP puede utilizar JavaScript Object Notation<sup>1</sup>(JSON) sobre WebSocket<sup>2</sup> para la comunicación entre el punto de carga y el sistema central. WebSocket permite la mensajería full duplex<sup>3</sup> sobre una sola

---

<sup>1</sup> JSON (Notación de objetos JavaScript) es un formato ligero de intercambio de datos.

<sup>2</sup> WebSockets es un protocolo de red basado en TCP que establece cómo deben intercambiarse datos entre redes.

<sup>3</sup> Full Duplex es un término que describe la transmisión y recepción de datos simultáneas a través de un canal.



conexión TCP<sup>4</sup> entre el cliente y el servidor. En OCPP que utiliza JSON sobre WebSocket, el punto de carga actúa como un cliente WebSocket, mientras que el sistema central es el servidor WebSocket (Priyasta et al., 2023)

En la Tabla 1, se muestran las características y los mensajes asociados agrupados por perfiles.

Tabla 1 Perfiles de OCPP (Fuente: (Open Charge Alliance, 2017))

Perfiles	Descripción
Core	Conjunto central de funciones y mensajes que se deben implementar para asegurar la interoperabilidad básica entre un punto de carga y un sistema central de gestión.
Gestión de firmware	Capacidad de gestionar y actualizar el firmware de un punto de carga de manera remota.
Gestión de la lista de autorizaciones locales	Permite a los operadores de estaciones de carga gestionar una lista local de autorizaciones para permitir o denegar el acceso a determinados vehículos eléctricos (EV) en una estación de carga específica.
Reservación	Capacidad de reservar un punto de carga para un vehículo eléctrico en un momento específico.
Carga inteligente	Capacidad de implementar y gestionar estrategias de carga inteligente.
Activación remota	Capacidad de iniciar una acción o evento en un punto de carga de forma remota desde el sistema central.

Esta investigación se centrará en la implementación del perfil “Core”. Define funcionalidades básicas y esenciales del protocolo OCPP, permitiendo la comunicación estándar entre las estaciones de carga y los sistemas de gestión de carga. Facilita los comandos y mensajes necesarios para establecer la conexión, iniciar y detener la carga, comunicar el estado de la carga y gestionar las transacciones de carga. Es el perfil mínimo requerido para la interoperabilidad básica entre los diferentes componentes del sistema de carga.

---

<sup>4</sup>TCP es un protocolo que establece conexiones entre dos puntos finales de comunicación.

Ocpp proporciona un conjunto de mensajes para operaciones iniciadas por el punto de carga, como **BootNotification**, **StatusNotification**, **Authorize**, **StartTransaction** y **StopTransaction**, así como un conjunto de mensajes para operaciones iniciadas por el sistema central, como **RemoteStartTransaction** y **RemoteStopTransaction**.

#### Operaciones iniciadas por el punto de carga:

- **Authorize:** contiene un token que posee un identificador que se utilizará para la autorización. Antes de que el propietario de un vehículo eléctrico pueda comenzar o detener la carga, el punto de carga debe autorizar la operación. El punto de carga deberá suministrar energía solo después de la autorización. Al detener una transacción, el punto de carga deberá enviar un *“Authorize.req”* solo cuando el identificador utilizado para detener la transacción sea diferente del identificador que inició la transacción (Open Charge Alliance, 2017).
- **Boot Notification:** después del arranque, un punto de carga deberá enviar una solicitud al sistema central con información sobre su configuración. El sistema central deberá responder para indicar si aceptará el punto de carga. El punto de carga deberá enviar un mensaje *“BootNotification.req”* cada vez que se encienda o reinicie. entre el encendido/reinicio físico y la finalización exitosa de un *“BootNotification”*, donde el sistema central devuelve aceptado o pendiente, el punto de carga no deberá enviar ninguna otra solicitud al sistema central (Open Charge Alliance, 2017).
- **Data Transfer:** cuando un punto de carga necesita enviar información al sistema central para una función no compatible con Ocpp, deberá utilizar el mensaje *“DataTransfer.req”* (Open Charge Alliance, 2017).
- **Status Notification:** un punto de carga envía una notificación al sistema central para informarle sobre un cambio de estado o un error dentro del punto de carga (Open Charge Alliance, 2017).
- **Heartbeat:** para informar al sistema central que un punto de carga sigue conectado, este envía un latido después de un intervalo de tiempo configurable. El punto de carga deberá enviar un mensaje *“Heartbeat.req”* para garantizar que el sistema central sepa que el punto de carga sigue activo (Open Charge Alliance, 2017).
- **Meter Values:** un punto de carga puede muestrear el medidor eléctrico u otro hardware de sensor/transductor para proporcionar información adicional sobre sus valores de medición. depende del punto de carga decidir cuándo enviará los valores del medidor. Esto se puede configurar utilizando el mensaje *“ChangeConfiguration.req”* para intervalos de adquisición de datos y especificar los datos a adquirir e informar (Open Charge Alliance, 2017).

- **Start Transaction:** el punto de carga deberá enviar un mensaje “*StartTransaction.req*” al sistema central para informar sobre una transacción que se ha iniciado. Si esta transacción finaliza una reserva, entonces el “*StartTransaction.req*” deberá contener el *reservationId*. Al recibir un mensaje “*StartTransaction.req*”, el sistema central debería responder con un mensaje “*StartTransaction.conf*”. Este mensaje de respuesta debe incluir un identificador de transacción y un valor de estado de autorización. El sistema central debe verificar la validez del identificador en el mensaje “*StartTransaction.req*”, porque el identificador podría haber sido autorizado localmente por el punto de carga utilizando información desactualizada (Open Charge Alliance, 2017).
- **Stop Transaction:** cuando se detiene una transacción, el punto de carga debe enviar un mensaje “*StopTransaction.req*”, notificando al sistema central que la transacción ha finalizado (Open Charge Alliance, 2017).

#### Operaciones iniciadas por el sistema central:

- **Change Availability:** el sistema central puede solicitar a un punto de carga que cambie su disponibilidad. Un punto de carga se considera disponible ("operativo") cuando está cargando o listo para cargar y se considera no disponible cuando no permite ninguna carga. El sistema central debe enviar un mensaje “*ChangeAvailability.req*” para solicitar a un punto de carga que cambie su disponibilidad, puede cambiar la disponibilidad a disponible o no disponible (Open Charge Alliance, 2017).
- **Change Configuration:** el sistema central puede solicitar a un punto de carga que cambie los parámetros de configuración. Para lograr esto, el sistema central debe enviar un “*ChangeConfiguration.req*”. Esta solicitud contiene un par clave-valor, donde "clave" es el nombre del ajuste de configuración a cambiar y "valor" contiene el nuevo valor para el ajuste de configuración (Open Charge Alliance, 2017).
- **Clear Cache:** el sistema central puede solicitar a un punto de carga que borre su caché de autorización. el sistema central debe enviar un mensaje “*ClearCache.req*” para borrar la caché de autorización del punto de carga. Al recibir un mensaje “*ClearCache.req*”, el punto de carga debe responder con un mensaje “*ClearCache.conf*”. El mensaje de respuesta debe indicar si el punto de carga pudo borrar su caché de autorización (Open Charge Alliance, 2017).
- **Data Transfer:** si el sistema central necesita enviar información a un punto de carga para una función no compatible con OCPP, debe utilizar el mensaje “*DataTransfer.req*”. El comportamiento de esta operación es idéntico a la operación de transferencia de datos iniciada

por el punto de carga. Consulta la transferencia de datos para obtener detalles (Open Charge Alliance, 2017).

- **Get Configuration:** para obtener el valor de las configuraciones, el sistema central deberá enviar un mensaje “*GetConfiguration.req*” al punto de carga. Si la lista de claves en el mensaje de solicitud está vacía o falta (es opcional), el punto de carga deberá devolver una lista de todas las configuraciones en “*GetConfiguration.conf*”. De lo contrario, el punto de carga deberá devolver una lista de claves reconocidas y sus valores correspondientes y el estado de solo lectura. Las claves no reconocidas deberán colocarse en el mensaje de respuesta como parte del elemento de lista de claves desconocidas opcional de “*GetConfiguration.conf*” (Open Charge Alliance, 2017).
- **Remote Start Transaction:** el sistema central puede solicitar a un punto de carga que inicie una transacción enviando un “*RemoteStartTransaction.req*”. Al recibirlo, el punto de carga debe responder con “*RemoteStartTransaction.conf*” y un estado que indique si ha aceptado la solicitud y tratará de iniciar una transacción (Open Charge Alliance, 2017).
- **Remote Stop Transaction:** el sistema central puede solicitar a un punto de carga que detenga una transacción enviando un “*RemoteStopTransaction.req*” al punto de carga con el identificador de la transacción. El punto de carga debe responder con “*RemoteStopTransaction.conf*” y un estado que indique si ha aceptado la solicitud y si hay una transacción con el *transactionId* proporcionado en curso y se detendrá. Esta solicitud remota para detener una transacción es equivalente a una acción local para detener una transacción. Por lo tanto, la transacción debe detenerse. el punto de carga debe enviar un “*StopTransaction.req*” y, si corresponde, desbloquear el conector (Open Charge Alliance, 2017).
- **Reset:** el sistema central debe enviar un mensaje “*Reset.req*” para solicitar que un punto de carga se reinicie. El sistema central puede solicitar un reinicio duro o suave. Tras recibir un mensaje “*Reset.req*”, el punto de carga debe responder con un mensaje “*Reset.conf*”. El mensaje de respuesta debe indicar si el punto de carga intentará reiniciarse (Open Charge Alliance, 2017).
- **Unlock Connector:** el sistema central puede solicitar a un punto de carga que desbloquee un conector. Para hacerlo, el sistema central debe enviar un mensaje “*UnlockConnector.req*”. El propósito de este mensaje es ayudar a los conductores de vehículos eléctricos que tienen problemas para desenchufar su cable del punto de carga en caso de mal funcionamiento del mecanismo de retención del cable del conector (Open Charge Alliance, 2017).

- **Trigger Message:** la solicitud “*TriggerMessage*” permite al sistema central solicitar al punto de carga que envíe mensajes iniciados por el punto de carga. En la solicitud, el sistema central indica qué mensaje desea recibir. Para cada mensaje solicitado, el sistema central puede indicar opcionalmente a qué conector se aplica esta solicitud (Open Charge Alliance, 2017).

### **Sesión de carga**

Una sesión de carga se inicia cuando se produce la primera interacción con el usuario o vehículo eléctrico. Esto puede ser un pase de tarjeta, inicio remoto de transacción, conexión de cable y/o vehículo eléctrico, detector de ocupación de estacionamiento (Open Charge Alliance, 2017).

Un punto de carga es el sistema físico donde se puede cargar un vehículo eléctrico, este tiene uno o más conectores (Open Charge Alliance, 2017). El conector es una toma de corriente operada y gestionada de forma independiente en un punto de carga (Open Charge Alliance, 2017).

### **1.2 Estudio de sistemas para la gestión de puntos de carga internacionales.**

**ChargeLab** es un sistema operativo para cargadores de vehículos eléctricos. De software de abierto, interoperable e independiente del hardware, incluye funciones para cada caso de uso. Registra y conecta cargadores en minutos con herramientas de autoservicio. Obtiene datos de carga históricos. A través del software, brinda soluciones de infraestructura para vehículos eléctricos más flexibles de la industria. Administra cualquier cargador OCPP (ChargeLab Inc, 2023).

**Ampcontrol** es un software de carga y gestión de energía basado en la nube que se conecta a cualquier cargador de vehículos eléctricos mediante OCPP. El sistema utiliza datos en tiempo real para monitorear el rendimiento del cargador, garantizar que los vehículos salgan a tiempo y reducir los costos totales de energía. El sistema de software de Ampcontrol se especializa en algoritmos de optimización que ayudan a los usuarios a garantizar una operación de carga confiable y eficiente. Las capacidades de gestión de energía de Ampcontrol permiten a los clientes gestionar el consumo de energía del cargador, así como la energía del sitio. Con la carga inteligente, los usuarios pueden optimizar las operaciones del sitio y también recibir señales de servicios públicos para la respuesta a la demanda o del vehículo a la red (Ampcontrol Technologies, 2023).

**Mobility Smart Manager** es una plataforma avanzada de gestión remota diseñada para estaciones de carga de vehículos eléctricos. Este software integral y multifuncional facilita la administración empresarial al permitir la conexión y desconexión de la carga de manera eficiente. Con esta solución, los gestores de estaciones de carga pueden tener un control total y acceder a información en tiempo real sobre los diversos puntos de recarga (adminetecnic, 2019).

Estas son sus características principales:

- Puntos de recarga: añadir y conectar cargadores de diferentes fabricantes.

- Conexión: en tiempo real con los cargadores a través de un mapa dinámico con indicador de estado.
- Potencia y consumo: configurar la potencia y precio de cada una de las tomas de corriente desde su centro de control.
- Vehículos eléctricos: gestión y control de los vehículos de los usuarios con sus tipologías de conector.
- Usuarios: gestión y control de todas las transacciones realizadas por el usuario.
- RFID: vinculación del consumo del punto de recarga con el usuario.

**CHARX manage** es un software escalable diseñado para optimizar el rendimiento de los postes de carga, aumentar su disponibilidad y facilitar una facturación precisa a través del protocolo Open Charge Point Protocol (OCPP). La gestión de cargas distribuye eficientemente la potencia de conexión disponible, asegurando que los vehículos eléctricos se carguen de acuerdo con sus necesidades y protegiendo la conexión doméstica contra sobrecargas de manera óptima.

Ventajas:

- La gestión de cargas integrada evita que se produzcan costosos fallos y garantiza la disponibilidad del parque de carga.
- Fácil puesta en marcha, configuración y monitorización del parque de carga mediante el navegador web.
- Manejo táctil intuitivo para el conductor del coche eléctrico en el terminal central.
- Orientado al futuro y flexible gracias a las licencias ampliables y a la posibilidad de añadir puntos de recarga nuevos de forma sencilla.
- Conexión flexible a los sistemas backend y de gestión de edificios.

Durante el proceso de carga, CHARX manage registra todos los datos de transacciones y energía relevantes, los guarda en una base de datos SQL local y los transmite al proveedor backend mediante OCPP(Phoenix Contact, 2023).

**Tridens EV Charge** es un sistema de gestión de estaciones de recarga de vehículos eléctricos. Utiliza el Protocolo Abierto de Punto de Carga para permitir la monitorización 24/7 de estaciones de recarga individuales y del conjunto de la plataforma de recarga de VE. Proporciona actualizaciones en tiempo real sobre el estado de las estaciones, incluyendo el número total de estaciones conectadas, su estado actual y cualquier problema de comunicación. Ofrece interfaces API para sistemas de planificación de recursos empresariales y gestión de tarifas. Gestiona el equilibrio de carga entre diferentes estaciones y ajusta dinámicamente la tasa de carga de cada estación, optimizando el rendimiento de la

infraestructura eléctrica. Permite la rápida incorporación de nuevas estaciones de recarga que sean compatibles con OCPP, facilitando la expansión del parque de carga (Zorko, 2023).

## **1.2 Análisis comparativo de los sistemas de gestión.**

A partir del análisis de los sistemas estudiados anteriormente, se determinó que los sistemas internacionales presentan limitaciones significativas debido a su desarrollo sobre software propietario y a la restricción de servicios que impiden su uso dentro del territorio nacional. Estas limitaciones obstaculizan el objetivo de lograr la soberanía tecnológica que el país busca alcanzar.

Aunque las soluciones analizadas no resuelven por completo los problemas identificados en la investigación, han sido valiosas para identificar características esenciales que satisfacen las necesidades de los clientes. Entre estas características clave se destaca que siguen un estándar de comunicación que facilita la interacción entre estaciones de carga y sistemas de gestión. Además, implementan mecanismos de autenticación y autorización para garantizar el acceso exclusivo de usuarios autorizados a las estaciones de carga, asegurando operaciones como el inicio de sesiones de carga. Estas soluciones también permiten la gestión eficiente de sesiones de carga, con la capacidad de iniciar, detener y pausar sesiones. Además, facilitan el monitoreo y control de los conectores en las estaciones de carga. Del sistema ChargeLab se toma como referencia la obtención de datos de carga históricos.

## **1.3 Metodología de desarrollo de software**

Las metodologías de desarrollo de software engloban una serie de métodos y técnicas organizativas aplicadas a la creación y diseño de software informático. Estos métodos persiguen organizar de manera eficiente a los equipos implicados para que desarrollen con éxito sus funciones teniendo en cuenta aspectos como la dificultad del proyecto, la planificación, el presupuesto, los costes, los profesionales disponibles, el lenguaje que se va a utilizar («Metodologías de desarrollo de software», 2022). Se puede decir que la metodología de desarrollo puede entenderse como un conjunto de pasos y prácticas organizadas que guían la construcción del software. Su propósito es establecer una secuencia lógica de actividades, empleando métodos, modelos y herramientas, así como gestionando las habilidades del equipo de desarrollo. Este enfoque busca contribuir a la calidad final del producto, asegurando que todas las tareas se realicen de manera ordenada y eficiente.

Para el desarrollo de la solución, se usará la variación de la metodología de Proceso Unificado Ágil (AUP por sus siglas en inglés) siendo esta la elección apropiada debido a la compatibilidad con la metodología utilizada por el proyecto al que contribuye.

### **1.3.1 Metodología de desarrollo de software variación de AUP para la UCI**

La Universidad de las Ciencias Informáticas (UCI) desarrolló una versión de la metodología de desarrollo de software AUP (Proceso Ágil Unificado), con el fin de crear una metodología que se adapte al ciclo de vida definido por la actividad productiva de la universidad. Esta versión decide mantener para el ciclo de vida de los proyectos la fase de Inicio, pero modificando el objetivo de la misma y se unifican las restantes fases de la metodología de desarrollo de software AUP en una sola, nombrada Ejecución y agregándose también una nueva fase llamada Cierre (Daynelis Brito Morales et al., 2019) Esta metodología cuenta con 3 fases:

- Inicio: se llevan a cabo las actividades relacionadas con la planeación del proyecto.
- Ejecución: se ejecutan las actividades requeridas para desarrollar el software.
- Cierre: se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Durante el ciclo de vida de los proyectos se definen 7 disciplinas:

- Modelado de negocio
- Requisitos
- Análisis y diseño
- Implementación
- Pruebas Interna
- Pruebas de liberación
- Pruebas de Aceptación

Dentro de la disciplina de requisito se definen 4 posibles escenarios para modelar el sistema.

Escenario No 1: proyectos que modelen el negocio con CUN (Caso de Uso del Negocio) solo pueden modelar el sistema con CUS (Caso de Uso del Sistema)

Escenario No 2: proyectos que modelen el negocio con MC (Modelo Conceptual) solo pueden modelar el sistema con CUS (Caso de Uso del Sistema)

Escenario No 3: proyectos que modelen el negocio con DPN (Descripción de Proceso de Negocio) solo pueden modelar el sistema con DRP (Descripción de Requisitos por Proceso)

Escenario No 4: proyectos que no modelen negocio solo pueden modelar el sistema con HU (Historia de Usuario)

La presente investigación se enfoca en el Escenario No 4.

El Escenario No 4 se destaca por la estrecha colaboración entre el cliente y el equipo de desarrollo. Esto es de vital importancia en nuestro proyecto, ya que el cliente desempeña un papel fundamental en la definición de los requisitos y en la validación de las soluciones propuestas. La participación activa del cliente garantiza que los detalles de los requisitos se capturen de manera efectiva y que el producto



final sea realmente lo que el cliente necesita. En proyectos en los que la comprensión precisa de los requisitos es crucial, este enfoque resulta esencial.

#### **1.4 Lenguajes y herramienta para el modelado de la solución**

En la presente investigación se emplea como lenguaje de modelado UML y como herramienta para el modelado Visual Paradigm, a continuación, una breve descripción de ambos:

##### **1.4.1 Lenguaje Unificado de Modelado (UML) v2**

El lenguaje Unificado de Modelado (UML por sus siglas en inglés de *Unified Modeling Language*) es un lenguaje de modelado estandarizado que consiste en un conjunto integrado de diagramas, implementado para ayudar a los desarrolladores de sistemas y software a especificar, visualizar, construir y documentar los artefactos de los sistemas de software, así como para el modelado de negocios y otros sistemas que no son de software (Visual Paradigm, 2022).

##### **1.4.2 Herramienta para el modelado de la solución**

La ingeniería de software asistida por computadora (CASE) es la implementación de herramientas y métodos facilitados por computadora en el desarrollo de software. CASE se utiliza para garantizar un software de alta calidad y libre de defectos, garantiza un enfoque disciplinado y controlado y ayuda a los diseñadores, desarrolladores, evaluadores, gerentes y otros a ver los hitos del proyecto durante el desarrollo, también puede ayudar como almacén de documentos relacionados con proyectos, como planes de negocios, requisitos y especificaciones de diseño. Una de las principales ventajas de utilizar CASE es la entrega del producto final, que es más probable que cumpla con los requisitos del mundo real, ya que garantiza que los clientes sigan siendo parte del proceso.

##### **1.4.3 Visual Paradigm v8.0**

Visual Paradigm es una poderosa herramienta de diseño y gestión de sistemas de TI, multiplataforma y fácil de usar. Visual Paradigm proporciona a los desarrolladores de software una plataforma de desarrollo de vanguardia para construir aplicaciones de calidad más rápidas, mejores y más económicas. Facilita una excelente interoperabilidad con otras herramientas CASE y la mayoría de los IDE líderes, lo que mejora todo su proceso de desarrollo de Modelo-Código-Implementación en esta solución integral de compra en un solo lugar (Visual Paradigm, 2023).

Con lo expuesto anteriormente y teniendo en cuenta que la UCI posee la licencia académica para el uso de Visual Paradigm v 8.0, se considera la herramienta idónea para el desarrollo del proyecto. La herramienta posee un repertorio de funcionalidades muy completos para el desarrollo de software, yendo desde el análisis y diseño hasta la generación de código fuente y documentación.

#### **1.5 Tecnologías de implementación**

Para el desarrollo de la propuesta de solución se hace necesario la identificación del marco de trabajo y el entorno de desarrollo que se emplearán para la realización del servicio web de la gestión de estaciones de carga. Se ha elegido el lenguaje de programación Java, dado que la OCA ha desarrollado una librería en este lenguaje para la implementación del estándar OCPP.

### **1.5.1 Lenguajes de programación**

Un lenguaje de programación es un conjunto de instrucciones y sintaxis utilizadas para crear programas de software (SanghpriyaGautam, 2023).

**Java v17:** es completamente orientado a objetos. La programación orientada a objetos (OOP, por sus siglas en inglés) ayuda a tratar aplicaciones del mundo real. La inclusión de la herencia, el polimorfismo, la abstracción y la encapsulación hace que un programa sea orientado a objetos. Java utiliza su propio entorno de ejecución, por lo que las aplicaciones Java son seguras. Tiene una gestión sólida de la memoria y elimina automáticamente objetos que no se utilizan (Khoirom et al., 2020).

#### **Ventajas de Java:**

- Sencillez: es un lenguaje claro, de tipado fuerte y tiene expectativas estrictas que guían a los aprendices a pensar de la manera correcta.
- Facilidad de uso: es fácil de usar, escribir, compilar, depurar y aprender.
- Independencia de plataforma: es independiente de la plataforma, un lenguaje distribuido que admite la programación multi-hilo y proporciona Recolección Automática de Basura, entre otros beneficios.
- Programación orientada a objetos (OOP): El uso de OOP en Java permite establecer programas estándar y código reutilizable.
- Seguridad: fue desarrollado teniendo en cuenta una plataforma segura. Cuenta con un Administrador de Seguridad para cada aplicación donde se establecen las reglas de acceso a las clases.
- Librerías y marcos de trabajo bien probados: tiene una gran cantidad de librerías y marcos de trabajo bien probados que facilitan el desarrollo de aplicaciones.
- Costos de mantenimiento bajos: es relativamente económico de mantener, ya que no depende de un hardware específico (Khoirom et al., 2020).

#### **Desventajas de Java:**

- Almacenamiento en lugar de copias de seguridad: Java se enfoca particularmente en el almacenamiento de datos y no en la copia de seguridad de los mismos.
- Gestión de memoria costosa: Java requiere un espacio de memoria considerable, lo que puede hacer que la gestión de la memoria sea costosa.

- Código verbose: Los códigos Java tienden a ser verbosos. Se enfoca en ser más manejable, pero a veces se traduce en códigos extremadamente complejos y una gran cantidad de información (Khoirom et al., 2020).

### 1.5.2 Marco de trabajo

Un marco de trabajo es una estructura de archivos y utilidades que aceleran la programación de una aplicación informática, que provee una metodología de trabajo que sistematiza y facilita la generación de formularios, funciones y módulos de uso común, permitiendo al desarrollador dedicar su atención hacia los aspectos específicos de cada aplicación.

**Spring Boot:** es una extensión de *Spring Framework*. El objetivo principal de *Spring Boot* es crear rápidamente aplicaciones basadas en *Spring* sin requerir que los desarrolladores escriban la misma configuración repetitiva una y otra vez. Está construido sobre *Spring Framework* y forma parte integral de este. Actualmente es el más utilizado dentro del conjunto de los marcos de trabajo, al igual que dentro del mundo Java. El objetivo principal de *Spring Boot* es simplificar el desarrollo de aplicaciones por medio de la autogestión de un gran número de configuraciones, tareas y componentes que son necesarios para la ejecución de una aplicación. *Spring Boot* ofrece a los desarrolladores un conjunto de configuraciones automáticas por defecto que hacen que una aplicación web pueda desarrollarse y desplegarse de manera rápida, sencilla y a la vez segura, realizando una cantidad de pasos mínimas (Ramírez Pérez, 2020).

Spring Boot también se destaca por lo siguiente:

- Posee todas las características de Spring Framework.
- Implementa el principio de diseño de Abierto-Cerrado, donde esta es cerrada a modificación, pero abierto a extensión cuando se desee tener un mayor control de sus componentes.
- Posee servidores de aplicaciones y contenedores de Servlet embebido.
- Eliminar la necesidad de configurar la aplicación por medio de código o XML.
- Es una gran opción para crear aplicaciones basadas en microservicios.
- Ofrece un amplio soporte a diferentes tecnologías como son bases de datos relaciones y no relacionales, almacenamiento en caché, mensajería, procesamiento por lotes y más.
- Ofrece métricas de la aplicación por medio de una librería llamada Actuator, utilizada para exponer información importante sobre la aplicación que se encuentra en ejecución a través del monitoreo.

### 1.5.3 Entorno de desarrollo integrado

Entorno de desarrollo integrado por sus siglas en inglés IDE (*Integrated Development Environment*) es un programa que está básicamente compuesto por un conjunto de herramientas que son usadas por el programador. Los IDE fueron diseñados para proporcionar un único programa en el cual se pueda llevar a cabo todo el desarrollo de un sistema y aumentar la productividad de los programadores, y así proporcionar componentes necesarios para la creación de interfaces de usuarios (Ponce Briones, 2016).

### **Características de los IDE**

Los IDEs son aplicaciones de software que integran una gran cantidad de herramientas y servicios en un solo programa, además de permitir la creación, modificación, compilación, implementación y depuración de software (Ponce Briones, 2016). Otras características generales que presentan los IDEs son las siguientes:

- Son multiplataforma.
- Capaces de soportar diversos lenguajes de programación.
- Permite la integración con sistemas de control de versiones.
- Identificación del modelo de sintaxis en la cual se trabaja.
- Cuenta con extensiones y componentes para el IDE.
- Permite la integración con marcos de trabajo reconocidos.
- Depurador.
- Permite importar y exportar proyectos.
- Trabaja en diferentes idiomas.
- Contiene manual de usuarios y ayuda.

### **Ventajas de los IDE**

La utilización de IDE permite la creación de nuevos programas y/o aplicaciones que ayuden al desarrollador de software en sus tareas y por lo tanto otorgan las siguientes ventajas (Ponce Briones, 2016):

- Facilita la creación de las tareas básicas en el desarrollo (guardar los datos, compilar-enlazar-ejecutar los datos) los cuales se unen en la barra de herramientas del desarrollo.
- El código fuente que se muestra está compuesto con un formato acoplado al lenguaje de

programación correspondiente en el trabajo realizado.

- Permite al usuario escoger opciones de auto completado de código fuente, seleccionar métodos a usar, parámetros, entre otros lo cual hace más eficiente la forma de programar.
- Alerta mensajes de error de sintaxis, mientras se está desarrollando o se está ejecutando la compilación del programa.
- Facilita una lista de métodos sobre cómo realizar las correcciones de errores presentados en la compilación.
- Explica método para realizar la refactorización de archivos y códigos fuente y mejorar la curva de aprendizaje.
- Se crean proyectos para revisar los datos de los archivos de una forma gráfica y amigable (Ponce Briones, 2016).

**IntelliJ IDEA (2023.1.1)** es uno de los entornos de desarrollo integrado (IDE) más potentes y populares para Java. Es desarrollado y mantenido por JetBrains y está disponible como edición comunitaria y definitiva. Este IDE rico en características permite un desarrollo rápido y ayuda a mejorar la calidad del código (Idea IntelliJ - Introducción, s. f.).

IntelliJ IDEA proporciona finalización de código sensible al contexto y le ofrece sugerencias que solo son válidas para la posición actual del cursor, utiliza el aprendizaje automático para garantizar que la sugerencia más relevante esté en la parte superior de la lista, permite trabajar con elementos de código en otros idiomas integrados en su código. *AI Assistant* viene con una funcionalidad útil que puede simplificar sus tareas diarias. Ofrece chat de IA integrado y puede hacer cosas como escribir automáticamente comentarios de documentación, sugerir nombres y generar mensajes de confirmación. IntelliJ IDEA es muy bueno para verificar la calidad y validez de su código mediante inspecciones sobre la marcha (jetbrains, 2023).

IntelliJ IDEA Ultimate brinda soporte de primera clase para marcos y tecnologías líderes orientados al desarrollo de aplicaciones y microservicios modernos. Su IDE viene con asistencia dedicada para Spring Boot.

Para el marco Spring Boot, IntelliJ IDEA Ultimate presenta información de código inteligente, inspecciones, navegación de código instantánea y configuraciones de ejecución altamente personalizables. También ofrece herramientas integradas que le permiten ejecutar y probar aplicaciones Spring y trabajar con solicitudes HTTP y herramientas de bases de datos. También puede

obtener una vista agregada de las API de cliente y servidor utilizadas en su proyecto para los protocolos HTTP y WebSocket en la ventana de herramientas Endpoints (jetbrains, 2023).

#### **1.5.4 Sistema Gestor de Base de datos**

Un sistema de gestión de bases de datos (SGBD) es un software utilizado para gestionar, almacenar y recuperar bases de datos. Proporciona una interfaz que permite a los usuarios leer, crear, borrar y actualizar datos (Betania V, 2023).

Los SGBD optimizan la organización de los datos mediante una técnica de esquema de base de datos llamada normalización. Como resultado, las grandes tablas de datos se dividen en partes más pequeñas para minimizar las redundancias y dependencias (Betania V, 2023).

Los SGBD también admiten el acceso concurrente, que permite que varios usuarios interactúen con una base de datos al mismo tiempo, manteniendo la integridad de los datos (Betania V, 2023).

#### **PostgreSQL v14**

PostgreSQL es un potente sistema de base de datos relacional de objetos de código abierto que utiliza y amplía el lenguaje SQL combinado con muchas características que almacenan y escalan de forma segura las cargas de trabajo de datos más complicadas (The PostgreSQL Global Development Group, 2023).

PostgreSQL viene con muchas características destinadas a ayudar a los desarrolladores a crear aplicaciones, administradores para proteger la integridad de los datos y crear entornos tolerantes a fallas, y ayudarlo a administrar sus datos sin importar cuán grande o pequeño sea el conjunto de datos. Además de ser gratuito y de código abierto, PostgreSQL es altamente extensible. Por ejemplo, puede definir sus propios tipos de datos, crear funciones personalizadas, incluso escribir código desde diferentes lenguajes de programación sin volver a compilar su base de datos (The PostgreSQL Global Development Group, 2023).

#### **Conclusiones del capítulo**

- El análisis del marco teórico referente a los sistemas de gestión de estaciones de carga, así como el estudio de los principales conceptos asociados al problema planteado, permitieron establecer las bases para el desarrollo de la investigación y la comprensión de características inherentes al objeto de estudio.
- La caracterización y comparación de los sistemas homólogos estudiados, demostraron la necesidad de desarrollar un servicio web para la gestión de estaciones de carga.

- Para guiar el proceso de desarrollo de la propuesta solución se eligió como metodología de desarrollo de software AUP en su variante adaptada para la UCI. Se definieron herramientas y tecnologías necesarias para configurar un entorno de desarrollo.

## **CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN.**

El capítulo describe la propuesta de solución como resultado de la investigación. Tomando como guía la metodología seleccionada, se obtienen en la disciplina de requisitos los requisitos funcionales y no funcionales como una primera aproximación de las características que debe cumplir la propuesta de solución. Por otra parte, se analiza la arquitectura base sobre la cual se implementa el sistema. Se obtienen los artefactos correspondientes a la disciplina de Análisis y diseño, aplicando los patrones de diseño definidos como buenas prácticas durante el ciclo de desarrollo del software, así como el patrón arquitectónico que da soporte a la propuesta de solución.

### **2.1 Descripción de la propuesta de solución**

Se llevará a cabo la implementación de un servicio web destinado a la gestión de estaciones de carga. A través del protocolo OCPP, el servicio se encarga de supervisar y controlar diversos aspectos, como el estado operativo, la disponibilidad de conectores y el seguimiento de sesiones de carga. Además, para enriquecer su funcionalidad, el sistema integra una API REST, desde la cual se adquiere datos fundamentales. Esta API proporciona datos cruciales, como detalles de usuarios autorizados y tokens habilitados para carga. Adicionalmente, la interacción con la API REST se extiende al proceso de registro de nuevos puntos de carga, garantizando una incorporación fluida de datos.

#### **2.1.1 Modelo conceptual**

El modelo conceptual de un sistema software constituye una abstracción externa que describe mediante diagramas y notaciones el conocimiento que debe poseer una persona acerca de un sistema. En la figura 1 se representa el modelo conceptual del servicio, El cual está compuesto:



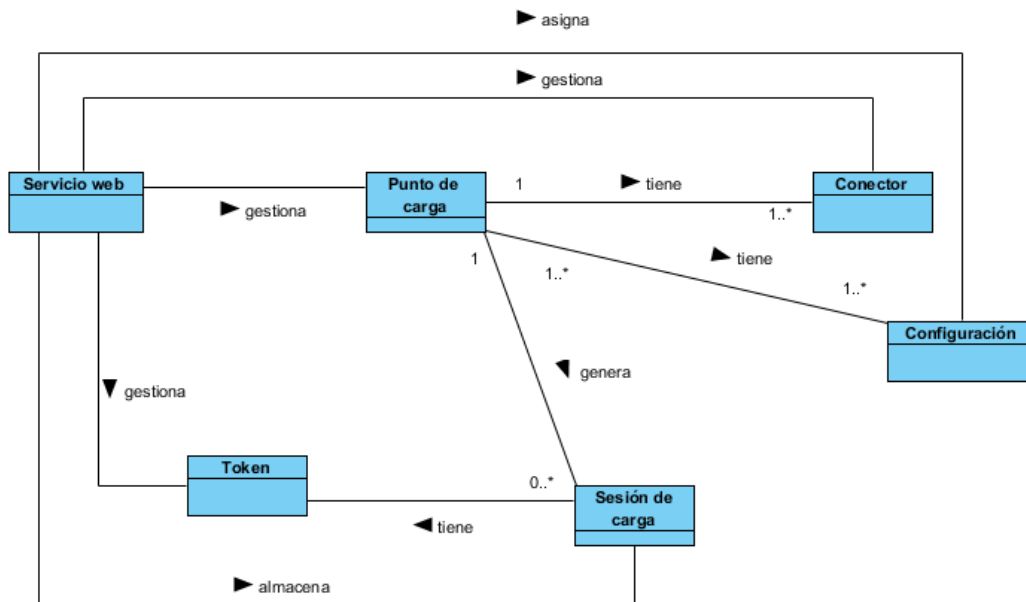


Figura 1 Modelo conceptual de la propuesta de solución (fuente: elaboración propia)

## 2.2 Requisitos de software

Los requisitos de un software son las descripciones de los servicios que un software debe proporcionar y las restricciones sobre su funcionamiento. Estos requisitos reflejan las necesidades de los clientes para un software que cumpla un determinado propósito, como controlar un dispositivo, realizar un pedido o encontrar información (Sommerville, 2016).

Estos requisitos pueden ser funcionales (describen las acciones que el software debe realizar) o no funcionales (especifican atributos de calidad como rendimiento, seguridad o usabilidad). Los requisitos de software son la base para el diseño, desarrollo y prueba del software (Sommerville, 2016).

### 2.2.1 Requisitos funcionales

Los requisitos funcionales de un sistema describen lo que el sistema debería hacer. Estos requisitos dependen del tipo de software que se está desarrollando, de los usuarios esperados del software y del enfoque general que adopta la organización al redactar requisitos. Cuando se expresan como requisitos de usuario, los requisitos funcionales deben estar escritos en lenguaje natural para que los usuarios y gerentes del sistema puedan entenderlos. Los requisitos funcionales del sistema amplían los requisitos

## CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

de usuario y están escritos para los desarrolladores del sistema. Deben describir las funciones del sistema, sus entradas y salidas, y las excepciones en detalle (Sommerville, 2016).

No	Requerimiento	Descripción
<b>RF1.</b>	Autenticar usuario.	Acción donde el usuario se va a autenticar en el sistema a través del servicio web.
<b>RF2.</b>	Registrar usuario.	Acción donde se registra un usuario en el sistema a través del servicio web.
<b>RF3.</b>	Eliminar usuario.	Acción donde se elimina un usuario en el sistema a través del servicio web.
<b>RF4.</b>	Actualizar usuario.	Acción donde se actualiza la información del usuario en el sistema a través del servicio web.
<b>RF5.</b>	Obtener usuario.	Acción donde se obtiene toda la información de un usuario registrado en el sistema a través del servicio web.
<b>RF6.</b>	Obtener lista de usuarios.	Acción donde se obtiene todos los usuarios registrados en el sistema a través del servicio web.
<b>RF7.</b>	Registrar un punto de carga.	Acción donde se registra un punto de carga en el sistema a través del servicio web.
<b>RF8.</b>	Actualizar un punto de carga.	Acción donde se actualiza la información de un punto de carga en el sistema a través del servicio web y a través del protocolo OCPP.
<b>RF9.</b>	Eliminar un punto de carga.	Acción donde se elimina un punto de carga en el sistema a través del servicio web.
<b>RF10.</b>	Obtener un punto de carga.	Acción donde se obtiene toda la información de un punto de carga registrado en el sistema a través del servicio web.
<b>RF11.</b>	Obtener lista de puntos de carga.	Acción donde se obtiene todos los puntos de carga registrados en el sistema a través del servicio web.

*CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN*

<b>RF12.</b>	Registrar una sesión de carga.	Acción donde se registra una sesión de carga en el sistema a través del protocolo OCPP.
<b>RF13.</b>	Actualizar una sesión de carga.	Acción donde se actualiza la información de un punto de carga en el sistema a través del protocolo OCPP.
<b>RF14.</b>	Eliminar una sesión de carga.	Acción donde se elimina una sesión de carga en el sistema a través del servicio web.
<b>RF15.</b>	Obtener una sesión de carga.	Acción donde se obtiene toda la información de una sesión de carga registrada en el sistema a través del servicio web.
<b>RF16.</b>	Obtener lista de sesiones de carga.	Acción donde se obtiene todas las sesiones de carga registradas en el sistema a través del servicio web.
<b>RF17.</b>	Registrar un conector de un punto de carga.	Acción donde se registra un conector al registrar un punto de carga en el sistema.
<b>RF18.</b>	Actualizar un conector de un punto de carga.	Acción donde se actualiza la información de un conector en el sistema a través del protocolo OCPP.
<b>RF19.</b>	Eliminar un conector de un punto de carga.	Acción donde se va a eliminar un conector en el sistema cuando se actualiza la información de la propiedad cantidad de conectores de un punto de carga.
<b>RF20.</b>	Obtener un conector de un punto de carga.	Acción donde se obtiene toda la información de un conector registrado en el sistema a través del servicio web.
<b>RF21.</b>	Obtener lista de los conectores de un punto de carga.	Acción donde se obtiene todos los conectores registrados en el sistema a través del servicio web.
<b>RF22.</b>	Iniciar una transacción remotamente de un punto de carga.	Acción donde se inicia una transacción a petición de un agente que no sea el punto de carga.

*CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN*

<b>RF23.</b>	Detener una transacción remotamente de un punto de carga.	Acción donde se detiene una transacción a petición de un agente que no sea el punto de carga.
<b>RF24.</b>	Desbloquear conector remotamente de un punto de carga.	Acción donde se desbloquea un conector de un punto de carga a petición de un agente que no sea el punto de carga.
<b>RF25.</b>	Obtener la configuración de un punto de carga.	Acción donde se obtiene toda la información de la configuración de un punto de carga registrado en el sistema a través del servicio web.
<b>RF26.</b>	Actualizar la configuración de un punto de carga.	Acción donde se actualiza la información de la configuración de un punto de carga en el sistema a través del servicio web y el protocolo OCPP.
<b>RF27.</b>	Restablecer un punto de carga a su configuración inicial.	Acción donde se restablece la configuración inicial de un punto de carga a través del servicio web y a través del protocolo OCPP se le solicita al punto de carga restablecer la configuración inicial.
<b>RF28.</b>	Cambiar disponibilidad de un punto de carga.	Acción donde se cambia la disponibilidad de un punto de carga a través del servicio web y a través del protocolo OCPP se le solicita al punto de carga cambiar la disponibilidad.
<b>RF29.</b>	Procesar las notificaciones de arranque enviadas desde un punto de carga.	Acción donde el sistema atiende las notificaciones de arranque enviadas de un punto de carga a través del OCPP.
<b>RF30.</b>	Procesar el "Heartbeat" enviada desde un punto de carga.	Acción donde el sistema atiende las señales periódicas llamadas "Heartbeat" enviadas desde un punto de carga.

<b>RF31.</b>	Registrar token.	Acción donde se registra un token en el sistema a través del servicio web.
<b>RF32.</b>	Eliminar token.	Acción donde se elimina un token en el sistema a través del servicio web.
<b>RF33.</b>	Actualizar token.	Acción donde se actualiza la información del token en el sistema a través del servicio web.
<b>RF34.</b>	Obtener token.	Acción donde se obtiene toda la información de un token registrado en el sistema a través del servicio web.
<b>RF35.</b>	Obtener lista de tokens.	Acción donde se obtiene todos los tokens registrados en el sistema a través del servicio web.
<b>RF36.</b>	Autorizar un token que solicita cargar en un punto de carga.	Acción donde el sistema va a autorizar a un token que solicita cargar en un punto de carga.
<b>RF37.</b>	Denegar un token que solicita cargar en un punto de carga.	Acción donde el sistema va a denegar a un token que solicita cargar en un punto de carga.
<b>RF38.</b>	Generar log de todas las operaciones que haga el sistema.	Acción donde el sistema generará reportes de todos los eventos y operaciones ocurridas en el sistema en formato log.

Tabla 2 requisitos funcionales (fuente: elaboración propia).

### 2.2.2 Requisitos no funcionales

Los requisitos no funcionales, son requisitos que no están directamente relacionados con los servicios específicos que el sistema brinda a sus usuarios. Estos requisitos no funcionales generalmente especifican o limitan las características del sistema en su conjunto. Pueden estar relacionados con propiedades emergentes del sistema, como la confiabilidad, el tiempo de respuesta y el uso de memoria. Alternativamente, pueden definir restricciones en la implementación del sistema, como las capacidades de los dispositivos de entrada/salida o las representaciones de datos utilizadas en las interfaces con otros sistemas (Sommerville, 2016).

Los requisitos no funcionales establecen criterios y restricciones que afectan la forma en que el sistema debe ser diseñado, implementado y utilizado. En el caso específico del sistema en cuestión, se han identificado 10 requisitos no funcionales relevantes:

#### Restricciones del software

**RnF 1.** Los cargadores asociados al sistema deben soportar como versión mínima OCPP v1.6.

**Mantenibilidad**

**RnF 2.** La implementación del software es uniforme y las funcionalidades son comentadas para facilitar su mantenimiento.

**Rendimiento**

**RnF 3.** El tiempo de respuesta del servicio a las solicitudes del cliente no puede superar los 2s.

**Escalabilidad**

**RnF 4.** El sistema debe ser capaz de manejar un aumento en el número de solicitudes y la carga de trabajo sin degradar su rendimiento.

**Seguridad**

**RnF 5.** El sistema debe garantizar la protección de los datos confidenciales y la prevención de accesos no autorizados, mediante técnicas como autenticación, cifrado y control de accesos.

**Hardware**

**RnF 6.** El sistema debe ejecutarse en una computadora con un procesador dual, Core o superior.

**RnF 7.** El sistema debe ejecutarse en una computadora con una memoria RAM mínima de 4 Gigabytes.

**Portabilidad**

**RnF 8.** El sistema debe funcionar en los sistemas operativos (Windows y Linux).

**Restricciones de desarrollo**

**RnF 9.** El sistema se debe desarrollar en el marco de trabajo Spring Boot.

**RnF 10.** El sistema se debe implementar en el lenguaje de programación java.

**2.3 Historias de usuario**

Una historia de usuario es una explicación informal de una función de software, escrita desde la perspectiva del usuario final. Estas historias deben escribirse utilizando un lenguaje no técnico para brindar contexto al equipo de desarrollo (Asana, 2022).

*Tabla 3 historia de usuario Autenticar usuario (fuente: elaboración propia)*

<b>Número:</b> RF1	Requisito: Autenticar usuario.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

<b>Tiempo Real:</b> 4 días
<b>Descripción:</b> El servicio permite la autenticación de un usuario cuando este proporciona su nombre de usuario y contraseña. El servicio verifica la validez de la combinación de usuario y contraseña en la base de datos interna. Si la autenticación es exitosa, el usuario obtiene acceso autorizado a las funciones y características del servicio web.
<b>Observaciones:</b>

Tabla 4 historia de usuario Registrar usuario (fuente: elaboración propia)

<b>Número:</b> RF2	Requisito: Registrar usuario.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3 días
<b>Descripción:</b> El servicio permite registrar nuevos usuarios a través del servicio web. Durante este procedimiento, un usuario previamente registrado introduce las credenciales de otro usuario, que son el nombre de usuario y la contraseña.	
<b>Observaciones:</b> Es importante destacar que este usuario y contraseña deben ser diferentes de las credenciales de otros usuarios ya registrados.	

Tabla 5 historia de usuario Eliminar usuario (fuente: elaboración propia)

<b>Número:</b> RF3	Requisito: Eliminar usuario.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1

*CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN*

<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3 días
<b>Descripción:</b> El servicio ofrece la funcionalidad de eliminar un usuario mediante el servicio web. Para llevar a cabo este proceso, es suficiente con proporcionar el identificador único del usuario que se desea eliminar.	
<b>Observaciones:</b>	

*Tabla 6 historia de usuario Actualizar usuario (fuente: elaboración propia)*

<b>Número:</b> RF4	Requisito: Actualizar usuario.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 4 días
<b>Descripción:</b> El servicio facilita la actualización de información de usuarios mediante el servicio web. Para llevar a cabo este proceso, se requiere proporcionar el identificador único del usuario y los datos actualizados que se desean modificar.	
<b>Observaciones:</b> Es importante destacar que esta operación de actualización se realiza exclusivamente a través del servicio web.	

*Tabla 7 historia de usuario Obtener usuario (fuente: elaboración propia)*

<b>Número:</b> RF5	Requisito: Obtener usuario.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1



CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.90 días
<b>Descripción:</b> El servicio permite la obtención de información de usuarios a través del servicio web. Este proceso se realiza proporcionando el identificador único del usuario.	
<b>Observaciones:</b>	

*Tabla 8 historia de usuario Obtener lista de usuarios (fuente: elaboración propia)*

<b>Número:</b> RF6	Requisito: Obtener lista de usuarios.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.8 días
<b>Descripción:</b> El servicio posibilita la obtención de la lista completa de usuarios a través del servicio web. Para llevar a cabo este proceso, se realiza una solicitud específica al servicio web, que devuelve la información detallada de todos los usuarios registrados.	
<b>Observaciones:</b>	

*Tabla 9 historia de usuario Registrar un punto de carga (fuente: elaboración propia)*

<b>Número:</b> RF7	Requisito: Registrar un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días

*CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN*

<b>Descripción:</b> Para registrar un nuevo punto de carga, es necesario transmitir al servicio la información correspondiente al identificador y la cantidad de conectores del punto de carga a través del servicio web.
<b>Observaciones:</b>

*Tabla 10 historia de usuario Actualizar un punto de carga (fuente: elaboración propia)*

<b>Número:</b> RF8	Requisito: Actualizar un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> Para actualizar la información de un punto de carga, solo se puede modificar el identificador y la cantidad de conectores por el servicio web y la información restante por el protocolo OCPP al recibir los datos del punto de carga.	
<b>Observaciones:</b>	

*Tabla 11 historia de usuario Eliminar un punto de carga (fuente: elaboración propia)*

<b>Número:</b> RF9	Requisito: Eliminar un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.8 días

*CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN*

**Descripción:** El servicio ofrece la funcionalidad de eliminar un punto de carga mediante el servicio web. Para llevar a cabo este proceso, es suficiente con proporcionar el identificador único del punto de carga que se desea eliminar.

**Observaciones:**

*Tabla 12 historia de usuario Obtener un punto de carga (fuente: elaboración propia)*

<b>Número:</b> RF10	Requisito: Obtener un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 4 días
<b>Descripción:</b> Para obtener toda la información de un punto de carga registrado en el sistema, se debe realizar una solicitud al servicio web utilizando su identificador único.	
<b>Observaciones:</b>	

*Tabla 13 historia de usuario Obtener lista de puntos de carga (fuente: elaboración propia)*

<b>Número:</b> RF11	Requisito: Obtener lista de puntos de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 4 días
<b>Descripción:</b> Para obtener la lista de todos los puntos de carga registrados en el sistema se solicitará a través del servicio web.	

**Observaciones:**

Tabla 14 historia de usuario Registrar una sesión de carga (fuente: elaboración propia)

<b>Número:</b> RF12	Requisito: Registrar una sesión de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 4 días
<b>Descripción:</b> Para registrar una sesión de carga el punto de carga envía un mensaje "StartTransaction" al servicio donde contiene la información necesaria para poder registrar una sesión de carga a través del protocolo OCPP.	
<b>Observaciones:</b>	

Tabla 15 historia de usuario Actualizar una sesión de carga (fuente: elaboración propia)

<b>Número:</b> RF13	Requisito: Actualizar una sesión de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> Para actualizar una sesión de carga, el punto de carga envía un mensaje notificando que se detuvo la carga.	
<b>Observaciones:</b>	

**CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN**

*Tabla 16 historia de usuario Eliminar una sesión de carga (fuente: elaboración propia)*

<b>Número:</b> RF14	Requisito: Eliminar una sesión de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3 días
<b>Descripción:</b> El servicio ofrece la funcionalidad de eliminar una sesión de carga mediante el servicio web. Para llevar a cabo este proceso, es suficiente con proporcionar el identificador único de la sesión que se desea eliminar.	
<b>Observaciones:</b>	

*Tabla 17 historia de usuario Obtener una sesión de carga (fuente: elaboración propia)*

<b>Número:</b> RF15	Requisito: Obtener una sesión de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> Para obtener toda la información de una sesión de carga registrada en el sistema, se solicitará a través del servicio web usando su identificador.	
<b>Observaciones:</b>	

*Tabla 18 historia de usuario Obtener lista de sesiones de carga (fuente: elaboración propia)*

--	--

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

<b>Número:</b> RF16	Requisito: Obtener lista de sesiones de carga.	
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1	
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días	
	<b>Tiempo Real:</b> 3.9 días	
<b>Descripción:</b> Para obtener la lista de todas las sesiones de carga registradas en el sistema, se realizará una solicitud a través del servicio web.		
<b>Observaciones:</b>		

Tabla 19 historia de usuario Registrar un conector de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF17	Requisito: Registrar un conector de un punto de carga.	
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1	
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días	
	<b>Tiempo Real:</b> 3.9 días	
<b>Descripción:</b> El servicio permite registrar nuevos conectores. Para registrar un conector se debe registrar primero el punto de carga, en el cual se introduce la cantidad de conectores que contiene.		
<b>Observaciones:</b>		

Tabla 20 historia de usuario Actualizar un conector de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF18	Requisito: Actualizar un conector de un punto de carga.	
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1	

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> Un conector se va a actualizar cuando llega el mensaje “Notificación de estado” a través del protocolo OCPP.	
<b>Observaciones:</b>	

Tabla 21 historia de usuario Eliminar un conector de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF19	Requisito: Eliminar un conector de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio ofrece la funcionalidad de eliminar un conector mediante el servicio web. Para llevar a cabo este proceso, es suficiente con proporcionar el identificador único del conector que se desea eliminar.	
<b>Observaciones:</b>	

Tabla 22 historia de usuario Obtener un conector de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF20	Requisito: Obtener un conector de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

<b>Descripción:</b> Para obtener toda la información de un conector registrado en el sistema, se solicitará a través del servicio web usando su identificador.
<b>Observaciones:</b>

Tabla 23 historia de usuario Obtener lista de los conectores de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF21	Requisito: Obtener lista de los conectores de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> Para obtener la lista de todos los conectores registrados en el sistema se solicitará a través del servicio web usando el identificador del punto de carga.	
<b>Observaciones:</b>	

Tabla 24 historia de usuario Iniciar una transacción remotamente de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF22	Requisito: Iniciar una transacción remotamente de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio envía un mensaje a través del protocolo al punto de carga donde le solicita iniciar una transacción.	



CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

<b>Observaciones:</b>
-----------------------

Tabla 25 historia de usuario Detener una transacción remotamente de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF23	Requisito: Detener una transacción remotamente de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio envía un mensaje a través del protocolo al punto de carga donde le solicita detener la transacción que se está realizando.	
<b>Observaciones:</b>	

Tabla 26 historia de usuario Desbloquear conector remotamente de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF24	Requisito: Desbloquear conector remotamente de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio envía un mensaje a través del protocolo OCPP al punto de carga donde le solicita desbloquear el conector.	
<b>Observaciones:</b>	

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

Tabla 27 historia de usuario Obtener la configuración de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF25	Requisito: Obtener la configuración de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> Para obtener toda la información de la configuración de un punto de carga registrado en el sistema, se debe realizar una solicitud al servicio web utilizando su identificador único.	
<b>Observaciones:</b>	

Tabla 28 historia de usuario Actualizar la configuración de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF26	Requisito: Actualizar la configuración de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio permite la actualización de la configuración de un punto de carga a través del protocolo OCPP y solo se modificará el identificador por el servicio web si es requerido.	
<b>Observaciones:</b>	

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

Tabla 29 historia de usuario Restablecer un punto de carga a su configuración inicial (fuente: elaboración propia)

<b>Número:</b> RF27	Requisito: Restablecer un punto de carga a su configuración inicial.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> Para restablecer la configuración inicial de un punto de carga el servicio le envía un mensaje solicitando dicha acción a través del protocolo OCPP.	
<b>Observaciones:</b>	

Tabla 30 historia de usuario Cambiar disponibilidad de un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF28	Requisito: Cambiar disponibilidad de un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio solicitará a un punto de carga el cambio de su disponibilidad. Un punto de carga se considera disponible (“operativo”) cuando se encuentra cargando o listo para cargar. Un punto de carga se considera no disponible cuando no permite realizar ninguna carga. El servicio debe enviar un mensaje para solicitar que un punto de carga cambie su disponibilidad.	
<b>Observaciones:</b>	

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

Tabla 31 historia de usuario Procesar las notificaciones de arranque enviadas desde un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF29	Requisito: Procesar las notificaciones de arranque enviadas desde un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El punto de carga cuando se reinicia o enciende envía toda su información al servicio a través del protocolo OCPP, donde guarda toda la información en la base de datos.	
<b>Observaciones:</b>	

Tabla 32 historia de usuario Procesar el "Heartbeat" enviada desde un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF30	Requisito: Procesar el "Heartbeat" enviada desde un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> para que el servicio sepa que un punto de carga todavía está conectado, un punto de carga envía un latido después de un intervalo de tiempo configurable.  El punto de carga debe enviar un mensaje para garantizar que el servicio sepa que un punto de carga todavía está activo.	

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

El servicio debe responder a esta solicitud con la hora actual del Sistema Central.
<b>Observaciones:</b>

*Tabla 33 historia de usuario Registrar token (fuente: elaboración propia)*

<b>Número:</b> RF31	Requisito: Registrar token.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio permite registrar nuevos tokens a través del servicio web. Durante este procedimiento, un usuario previamente registrado introduce los datos del token.	
<b>Observaciones:</b>	

*Tabla 34 historia de usuario Eliminar token (fuente: elaboración propia)*

<b>Número:</b> RF32	Requisito: Eliminar token.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio ofrece la funcionalidad de eliminar un token mediante el servicio web. Para llevar a cabo este proceso, es suficiente con proporcionar el identificador único del token que se desea eliminar.	

**Observaciones:**

Tabla 35 historia de usuario Actualizar token (fuente: elaboración propia)

<b>Número:</b> RF33	Requisito: Actualizar token.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio permite actualizar la información de un token registrado en la base de datos a través del servicio web, el cual se le pasa el identificador del token y la información a modificar.	
<b>Observaciones:</b>	

Tabla 36 historia de usuario Obtener token (fuente: elaboración propia)

<b>Número:</b> RF34	Requisito: Obtener token.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio permite obtener toda la información de un token registrado en la base de datos a través del servicio web proporcionando solo el identificador.	
<b>Observaciones:</b>	

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

Tabla 37 historia de usuario Obtener lista de tokens (fuente: elaboración propia)

<b>Número:</b> RF35	Requisito: Obtener lista de tokens.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio permite obtener una lista de todos los tokens registrados en la base de datos, a través del servicio web.	
<b>Observaciones:</b>	

Tabla 38 historia de usuario Autorizar un token que solicita cargar en un punto de carga (fuente: elaboración propia)

<b>Número:</b> RF36	Requisito: Autorizar un token que solicita cargar en un punto de carga.
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días
	<b>Tiempo Real:</b> 3.9 días
<b>Descripción:</b> El servicio permitirá autorizar un token que solicita cargar en un punto de carga si este se encuentra ya registrado en la base de datos.	
<b>Observaciones:</b>	

Tabla 39 historia de usuario Denegar un token que solicita cargar en un punto de carga (fuente: elaboración propia)

--	--

<b>Número:</b> RF37	Requisito: Denegar un token que solicita cargar en un punto de carga.	
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1	
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días	
	<b>Tiempo Real:</b> 3.9 días	
<b>Descripción:</b> El servicio permitirá denegar un token que solicita cargar en un punto de carga si esta no se encuentra registrado en la base de datos.		
<b>Observaciones:</b>		

Tabla 40 historia de usuario Generar log de todas las operaciones que haga el sistema (fuente: elaboración propia)

<b>Número:</b> RF38	Requisito: Generar log de todas las operaciones que haga el sistema.	
<b>Programador:</b> Claudia Collazo Gumá	<b>Iteración Asignada:</b> 1	
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 3.98 días	
	<b>Tiempo Real:</b> 3.9 días	
<b>Descripción:</b> El servicio permitirá generar reportes de todos los eventos y operaciones ocurridas en el sistema en formato log.		
<b>Observaciones:</b>		

## 2.4 Arquitectura del software

La arquitectura de software se define como la estructura o estructuras del sistema que proporcionan una visión conceptual del mismo, incluyendo los componentes del sistema, sus relaciones y cómo interactúan entre sí para cumplir con los requisitos del sistema (Richards & Ford, 2020).



Patrón arquitectónico: son soluciones probadas y prácticas para problemas recurrentes en el diseño de sistemas de software. Proporcionan un marco para la toma de decisiones de diseño al ofrecer un conjunto de principios y directrices que los arquitectos de software pueden seguir para construir sistemas de software que sean fáciles de mantener, extender y escalar. Cada patrón arquitectónico tiene sus propias limitaciones y beneficios, y es importante comprenderlos para aplicarlos de manera efectiva en el diseño del sistema (Richards & Ford, 2020).

Los patrones y los estilos arquitectónicos se utilizan juntos en el diseño de sistemas de software para proporcionar una estructura general al sistema. El patrón arquitectónico propone una solución probada y práctica para un problema recurrente en el diseño de sistemas de software, y sirve como base para el diseño de la arquitectura del sistema (Richards & Ford, 2020). En el desarrollo de este trabajo se hace uso de una arquitectura por capas.

#### 2.4.2 Arquitectura en capas

Los componentes dentro del patrón de arquitectura en capas se organizan en capas horizontales, y cada capa desempeña una función específica dentro de la aplicación. Las comunicaciones entre estas capas obedecen a un flujo unidireccional, en el que cada capa se relaciona exclusivamente con la capa directamente inferior. Cada una de estas capas asume responsabilidades definidas y se concentra en sus tareas específicas, sin verse afectada por las capas superiores o inferiores (O'Reilly Media, 2023). En la siguiente figura se muestra la arquitectura diseñada para la propuesta de solución, la cual consta de tres capas:

**Capa de controlador:** esta capa actúa como la interfaz principal a través de la cual se interactúa con el sistema. Su función principal es recibir las solicitudes del punto de carga y del servicio web, procesarlas y dirigir las al lugar adecuado en la capa de servicios para su procesamiento.

**Capa de servicio:** contiene la lógica de negocio de la aplicación, donde se procesan las solicitudes que llegan desde la capa de controlador. Esta capa se encarga de llevar a cabo todas las operaciones de procesamiento, validaciones y toma de decisiones necesarias para cumplir con los requisitos funcionales de la aplicación. La capa de servicio actúa como un intermediario entre la capa de controlador y la capa de acceso a datos.

**Capa de acceso a datos:** en esta capa se gestionan todas las operaciones relacionadas con la persistencia de datos, incluye la recuperación y el almacenamiento de datos en una base de datos. La capa de acceso de datos se encarga de traducir las solicitudes de la capa de servicio en consultas específicas para la base de datos.

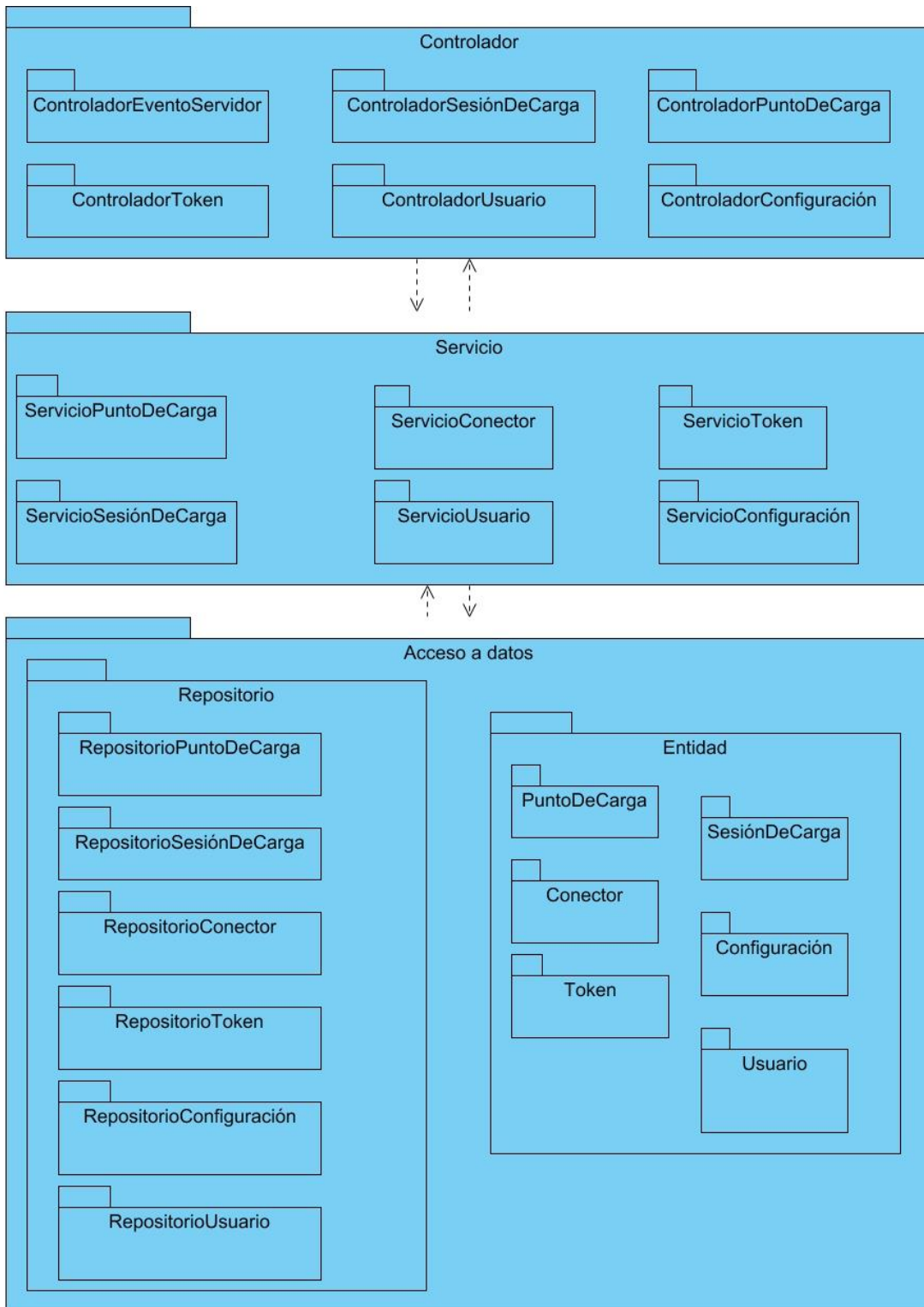


Figura 2 Diagrama de arquitectura por capas del sistema (fuente: elaboración propia).

## 2.5 Diagrama de clases

Los diagramas de clases son un componente esencial del proceso de modelado de objetos y se utilizan para representar la estructura estática de un sistema. Estos diagramas actúan como los planos de un sistema o subsistema, permitiendo modelar objetos, relaciones entre ellos y describir sus funcionalidades y servicios (IBM Corporation, 2021).

El diagrama de clases de la figura 3 responde a las funcionalidades de gestionar “token”.

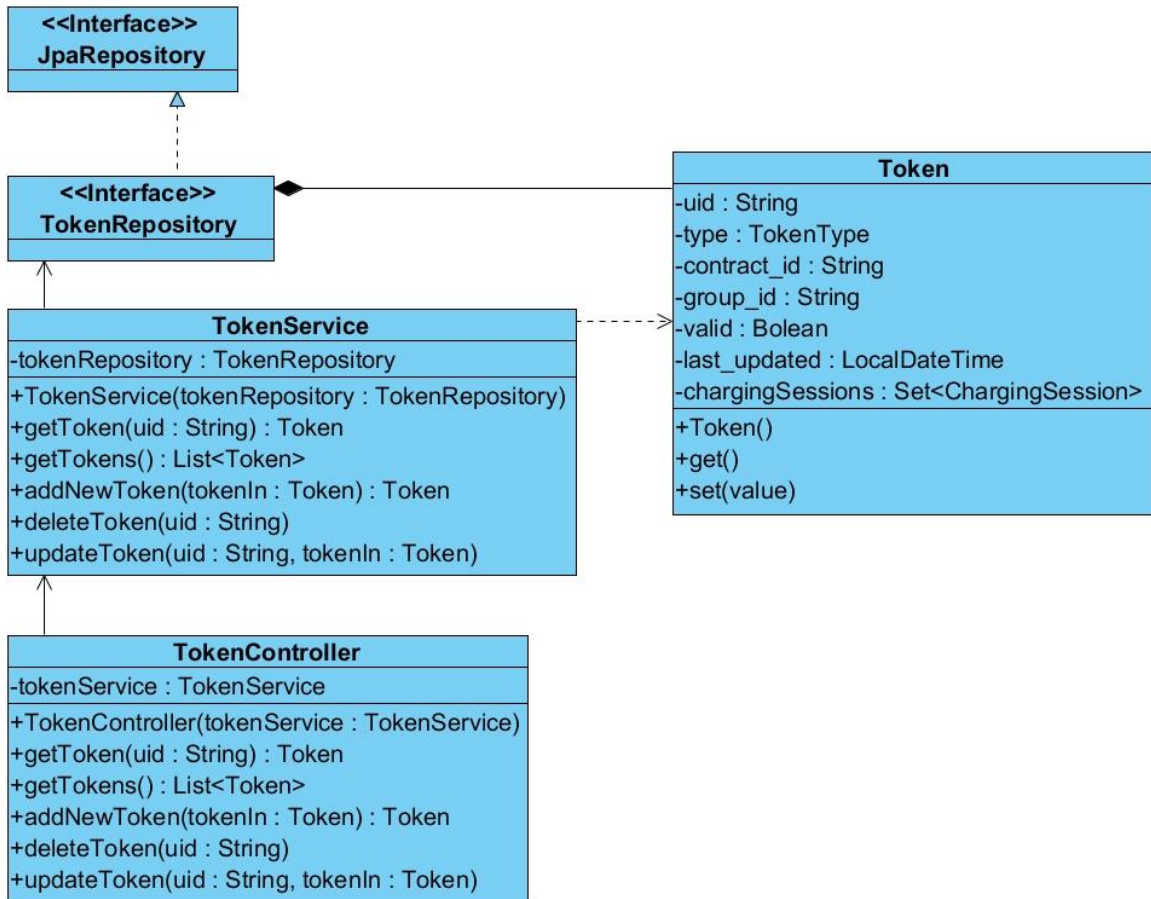


Figura 3 Diagrama de clases Gestionar token (fuente: elaboración propia).

El diagrama de clases de la figura 4 responde a las funcionalidades de gestionar la configuración de un punto de carga.

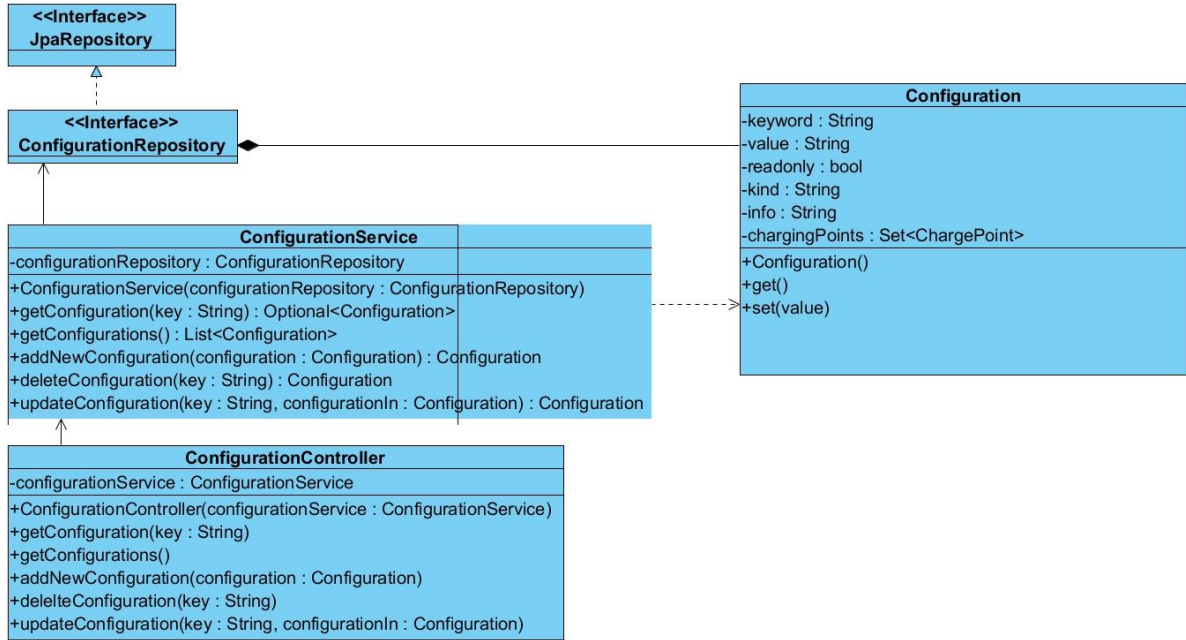


Figura 4 Diagrama de clases Gestionar configuración (fuente: elaboración propia).

El diagrama de clases de la figura 5 responde a las funcionalidades de gestionar las sesiones de un punto de carga.

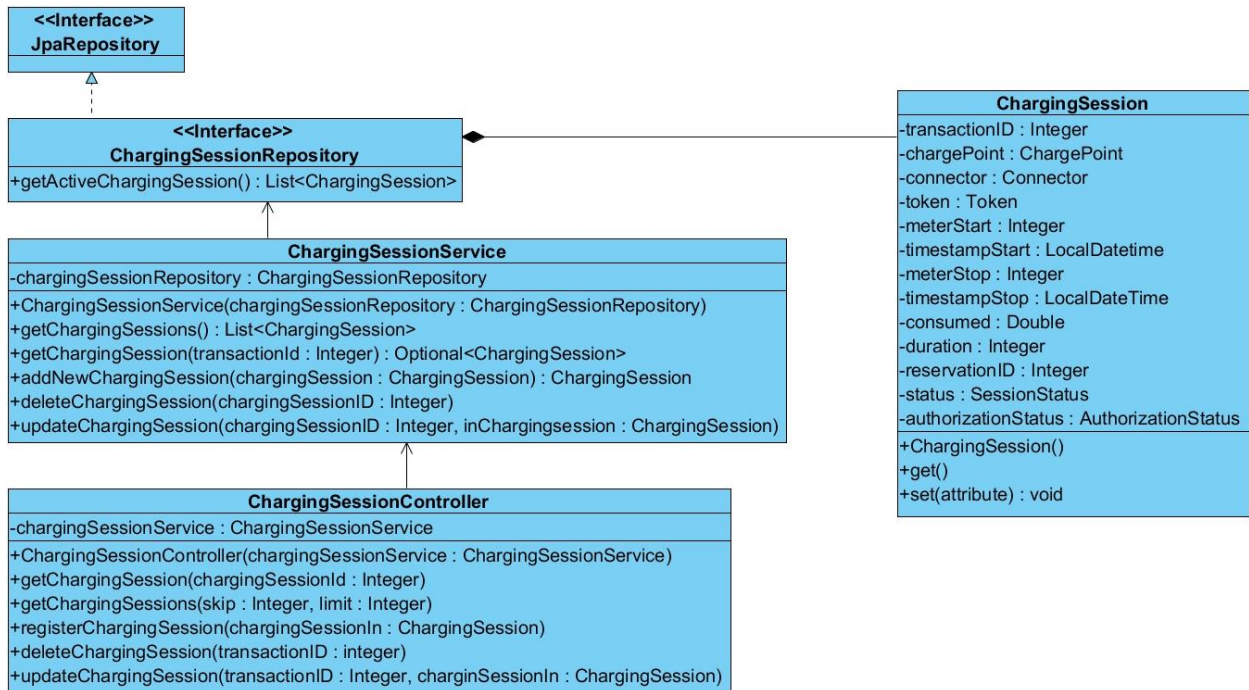


Figura 5 Diagrama de clases Gestionar sesiones de carga (fuente: elaboración propia).

El diagrama de clases de la figura 6 responde a las funcionalidades de interacción entre el servicio y el punto de carga.

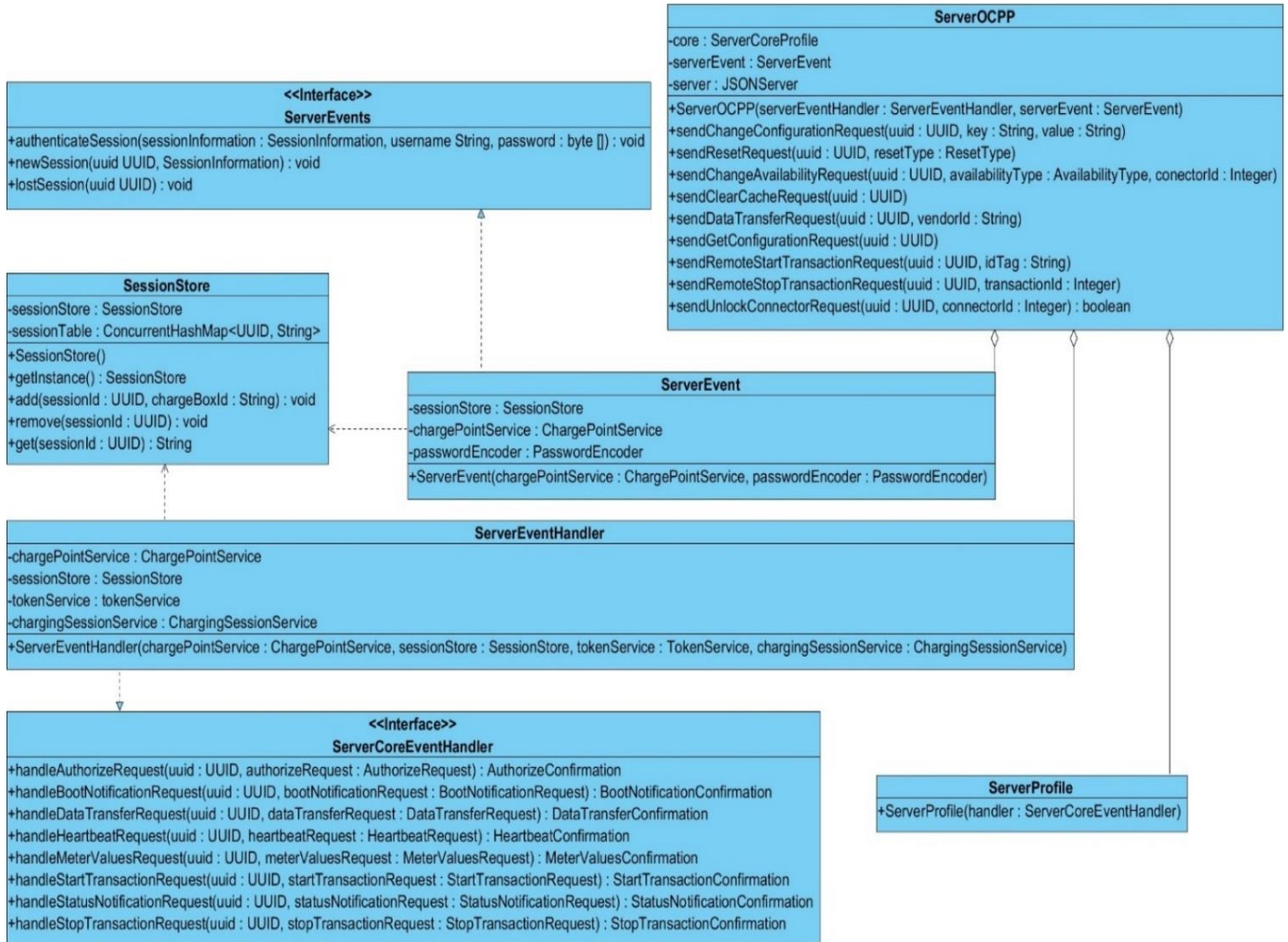


Figura 6 Diagrama de clases correspondiente a la interacción entre el sistema y el punto de carga (fuente: elaboración propia).

El diagrama de clases de la figura 7 responde a las funcionalidades de gestionar los conectores de un punto de carga.



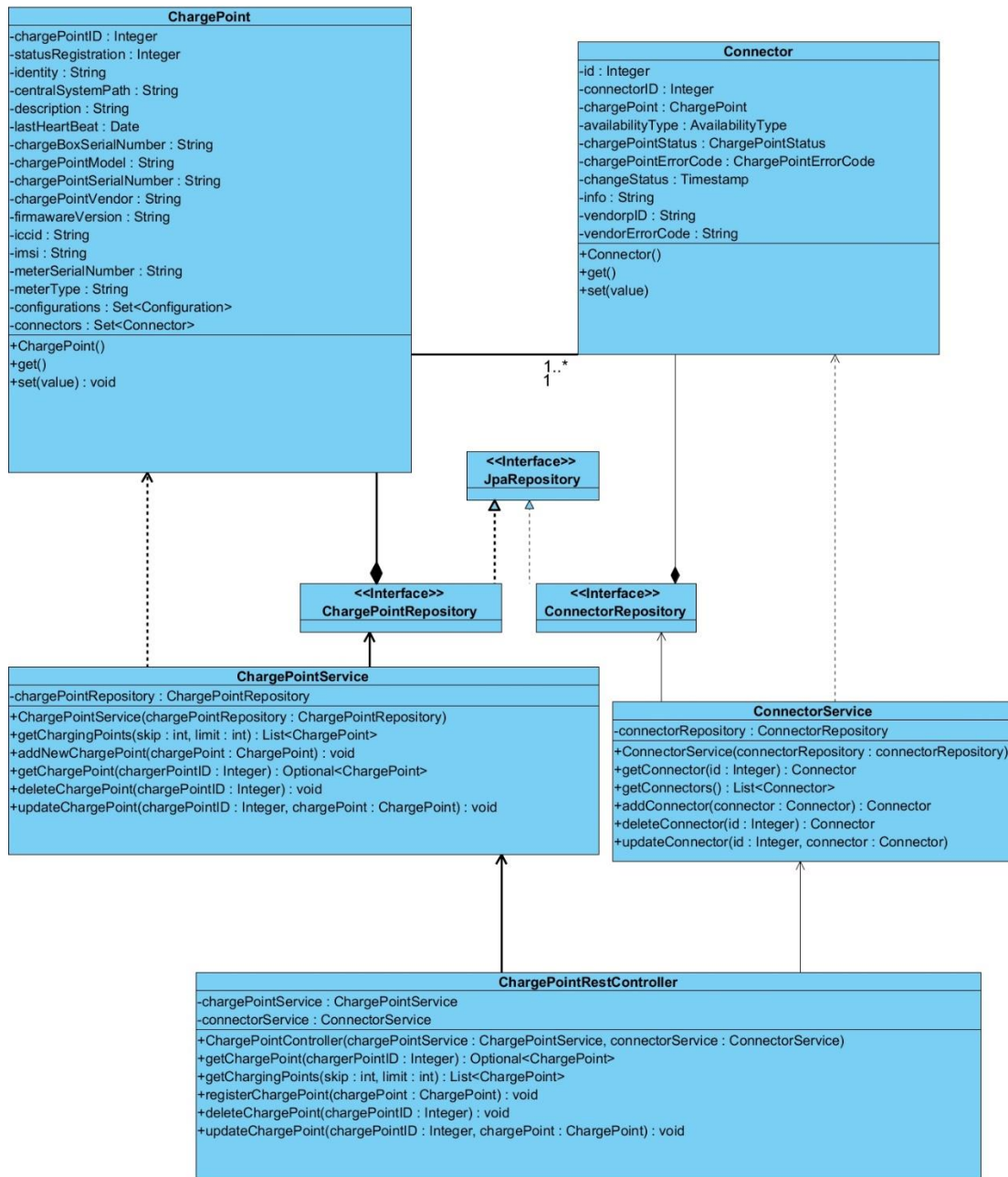


Figura 7 Diagrama de clases Gestionar conectores (fuente: elaboración propia).

## 2.6 Patrones de diseño

Los patrones de diseño son soluciones habituales a problemas que ocurren con frecuencia en el diseño de software. Son como planos prefabricados que se pueden personalizar para resolver un problema de diseño recurrente en tu código (Refactoring.Guru, 2014).

### 2.6.1 Patrones GRASP (General Responsibility Assignment Software Patterns)

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones.

**Controlador:** el objetivo de este patrón es asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas, o sea, delega la responsabilidad en otras clases con las que mantiene un modelo de alta cohesión. Este patrón queda evidenciado en la clase:

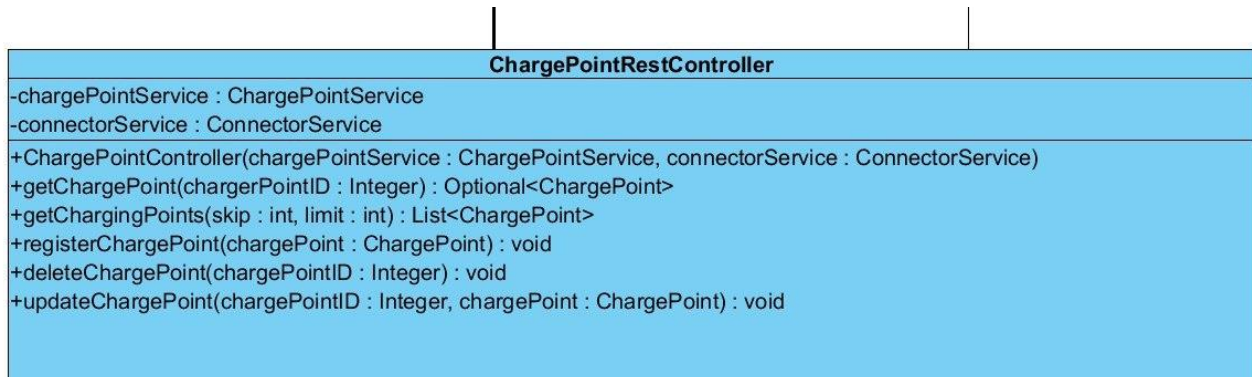


Figura 8 Patrón controlador (fuente: elaboración propia).

**Bajo acoplamiento:** Determina el nivel de dependencia de una clase con respecto a otras. Su uso potencia la reutilización, el mantenimiento y la mitigación de efectos a producirse en una clase al hacer cambios en otra.

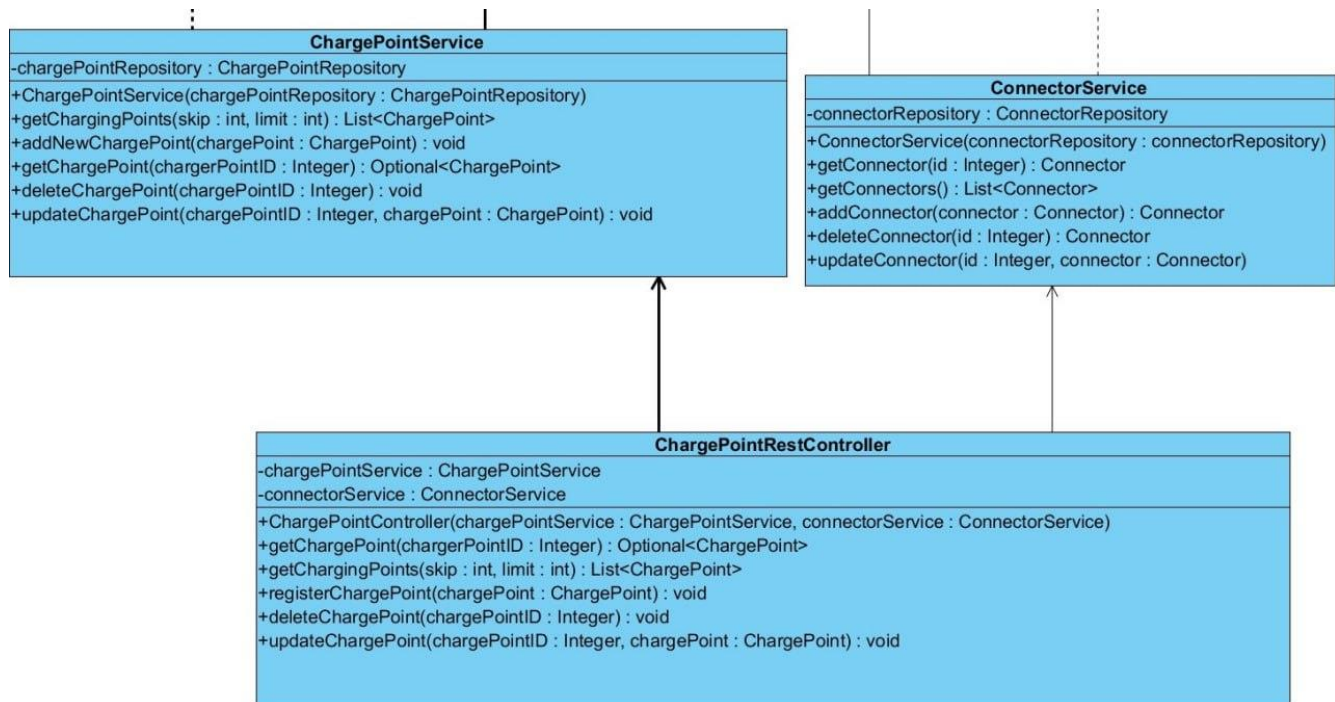


Figura 9 Patrón bajo acoplamiento (fuente: elaboración propia).

**Alta cohesión:** Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Significa que las clases del sistema tienen asignadas solo las responsabilidades que les corresponde y mantienen una estrecha relación con el resto de las clases.

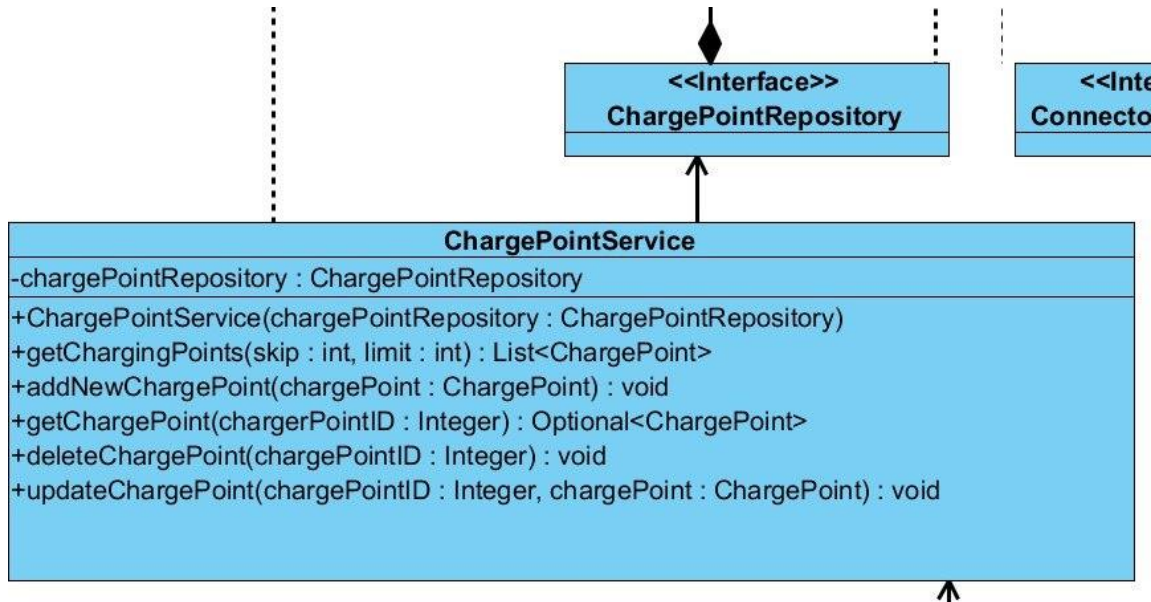


Figura 10 Patrón alta cohesión (fuente: elaboración propia).

**Experto:** se utiliza cuando se quiere asegurar que cada clase tenga la información que necesita y no dependa en exceso de otras clases para obtenerla.



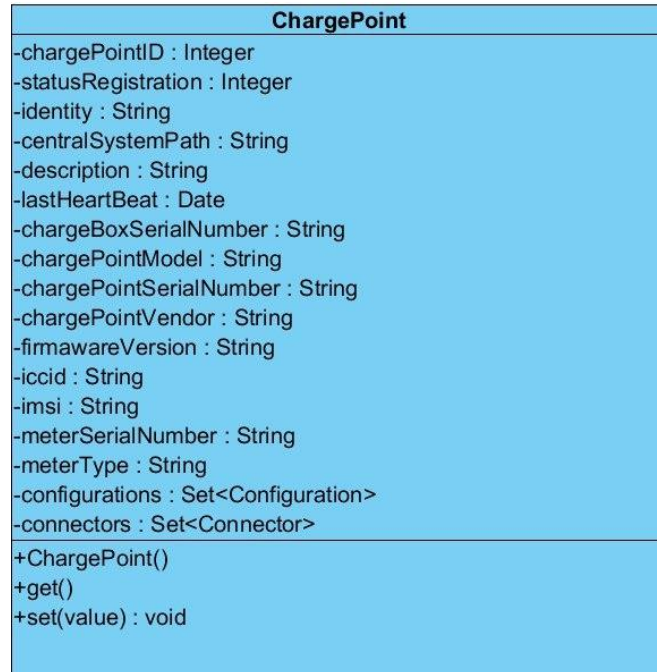


Figura 11 Patrón experto (fuente: elaboración propia)

### 2.6.2 PatronesGoF (Gang of Four)

**Adapter** es un patrón de diseño estructural que permite la colaboración entre objetos con interfaces incompatibles (Refactoring.Guru, 2014). En la solución se ve reflejada en la clase repositorio la que se encarga de transformar los datos provenientes de la base de datos a objetos de clases correspondientes.

**Singleton** es un patrón de diseño creacional que nos permite asegurarnos de que una clase tenga una única instancia, a la vez que proporciona un punto de acceso global a dicha instancia (Refactoring.Guru, 2014). En la solución se ve reflejada en la clase “*SessionStore*” ya que es necesario que se tenga una única instancia de esa clase.

### 2.7 Modelo de datos

El modelado de datos es una manera de estructurar y organizar los datos para que se puedan utilizar fácilmente por las bases de datos (tecnologías-información, 2018).

El modelado de datos es el proceso de analizar y definir todos los diferentes tipos de datos que su negocio recopila y produce, así como las relaciones entre esos bits de datos. Mediante el uso de texto, símbolos y diagramas, los conceptos de modelado de datos crean representaciones visuales de los datos que se capturan, almacenan y utilizan en su negocio. A medida que su empresa determina cómo y cuándo se utilizan los datos, el proceso de modelado de datos se transforma en un ejercicio de comprensión y clarificación de sus requisitos de datos (microsoft, 2023).

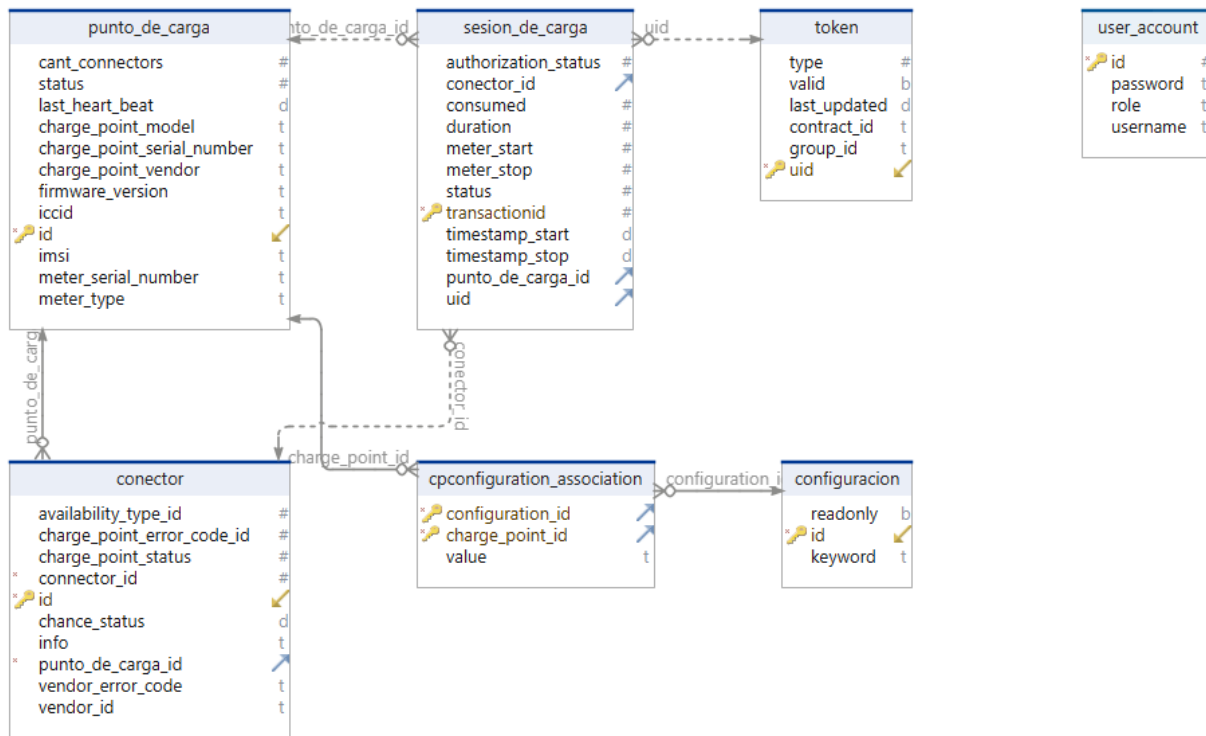


Figura 12 Modelo de datos (fuente: elaboración propia)

## 2.8 Diagrama de despliegue

Un Diagrama de Despliegue modela la arquitectura en tiempo de ejecución de un sistema. Esto muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos (Sparx Systems, 2023).

A continuación, se muestra el diagrama de despliegue correspondiente al sistema:

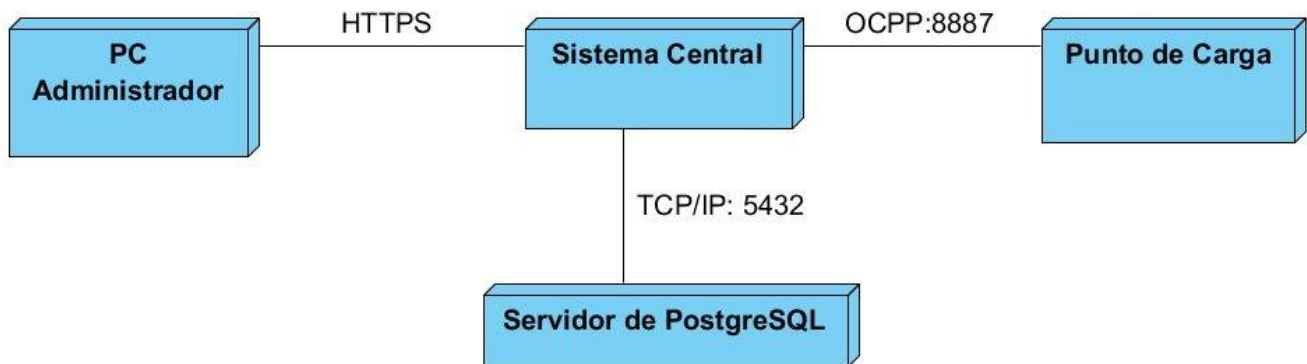


Figura 13 Diagrama de despliegue (fuente: elaboración propia)

La figura anterior representa el diagrama de despliegue para el sistema. El mismo estará compuesto por:

- PC Administrador: se va a comunicar con el sistema central a través del servicio web.
- Sistema Central: contiene toda la implementación del sistema, gestiona toda la comunicación OCPP y servicio web.
- Punto de Carga: representa los puntos de carga que se conectan al sistema a través del protocolo OCPP.
- Servidor de PostgreSQL: es el servidor de base de datos donde se almacena toda la información de la aplicación.

### Conclusiones del capítulo

- La descripción general de la propuesta de solución permitió adquirir una visión de los flujos de información y comprender el contexto de la solución propuesta.
- A través de la especificación de los requisitos e historias de usuario, se obtuvieron 38 requisitos funcionales, 10 requisitos no funcionales y 38 historias de usuario.
- La presentación clara y detallada de la arquitectura de software permitió comprender mejor cómo se debe construir la solución.
- El diagrama de clases del diseño permitió comprender mejor cómo se deben construir las clases y cómo se relacionan entre sí.
- La implementación de patrones de diseño mejora la escalabilidad, mantenibilidad y extensibilidad de la solución.

## *CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN*

- El modelo de datos es una herramienta útil que garantizó la calidad y la eficacia de la solución propuesta en cuanto a la gestión de datos.
- El diagrama de despliegue permitió obtener una representación visual de la infraestructura física y arquitectura del sistema.

## CAPÍTULO III: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

En el presente capítulo se evalúa el grado de calidad y fiabilidad de los resultados obtenidos en el desarrollo de la presente investigación. Para la validación de la solución se define una estrategia de prueba aplicando el nivel de unidad, con sus respectivos métodos y técnicas. Por otra parte, se realiza una verificación del cumplimiento del objetivo general de la investigación.

### 3.1 Pruebas de software

Las pruebas de software constituyen el conjunto de actividades realizadas para facilitar el descubrimiento y/o la evaluación de propiedades de uno o más elementos de prueba. Se considera un proceso crítico que puede demostrar la presencia de defectos, pero nunca demostrar la ausencia de ellos. Puede ser usado para prevenir los defectos, determinar la información necesaria y el nivel de calidad para la toma de decisiones (Figueredo, 2021).

### 3.2 Estrategia de pruebas

Una estrategia de prueba de software proporciona una guía que describe los pasos a seguir como parte de la prueba, cuándo se planifican y se llevan a cabo estos pasos, y cuánto esfuerzo, tiempo y recursos se requerirán. Por lo tanto, cualquier estrategia de prueba debe incorporar la planificación de las pruebas, el diseño de casos de prueba, la ejecución de pruebas y la recopilación y evaluación de los datos resultantes (Pressman & Maxim, 2015).

La estrategia diseñada queda de la siguiente manera:

Se evaluará el nivel de unidad, utilizando el método de Caja Blanca con la ayuda de la técnica de camino básico y se ejecutarán de manera automatizada haciendo uso de la herramienta JUnit5. También se realizará pruebas a nivel de integración, con el método de Caja Negra, empleando la herramienta Postman para evaluar el servicio web.

#### 3.2.1 Nivel unidad. Métodos de Caja Blanca

En este nivel, se crean casos de pruebas específicos para cada unidad o componente del sistema. Estos casos de prueba deben incluir tanto entradas válidas como inválidas y deben asegurar que cada unidad o componente esté funcionando correctamente y cumpliendo con sus requisitos.

Para la ejecución de estas pruebas se hará uso de la técnica de prueba de caja blanca camino básico, con la idea de derivar los casos de prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control. Para obtener el conjunto de caminos independientes se construirá el Grafo de Flujo asociado y se calculará su complejidad ciclomática. La automatización de las pruebas se llevará a cabo utilizando la herramienta JUnit5 que es un marco simple para escribir pruebas automatizadas (JUnit, 2021).

A continuación, se detallan los pasos que se utilizaron al aplicar la técnica de camino básico a la implementación del método “*sendUnlockConnectorRequest*” encargado de “Desbloquear conector remotamente de un punto de carga”:

**Paso 1:**

Elaborar el grafo de flujo con el objetivo de mostrar el flujo de control lógico. Este está compuesto por los siguientes elementos:

1. Nodos: círculos que representan uno o más declaraciones procedimentales (Pressman & Maxim, 2015).
2. Aristas o enlaces: flechas que representan el flujo de control y son análogas a las flechas del diagrama de flujo (Pressman & Maxim, 2015).
3. Regiones: áreas delimitadas por aristas y nodos (Pressman & Maxim, 2015).

```

boolean sendUnlockConnectorRequest(UUID uuid, Integer connectorId){
    UnlockConnectorRequest request = this.core.createUnlockConnectorRequest(connectorId); } 1

    try {
        CompletionStage<Confirmation> confirmationCompletionStage = this.server.send(uuid, request); } 2
        CompletableFuture<Confirmation> future = confirmationCompletionStage.toCompletableFuture(); } 3
        Confirmation confirmation = future.get();

        if (!(confirmation instanceof UnlockConnectorConfirmation)) } 4
            return false;

        UnlockConnectorConfirmation unlockConnectorConfirmation = (UnlockConnectorConfirmation) confirmation; } 5
        UnlockStatus unlockStatus = unlockConnectorConfirmation.getStatus();

        if (unlockStatus != UnlockStatus.Unlocked) } 6
            return false;
    }
    catch (Exception e){
        System.out.printf("%1$s: %2$s%n", e.getClass(), e.getMessage()); } 7
        return false;
    }

    return true; } 8
} } F
    
```

Figura 14 Implementación del método "sendUnlockConnectorRequest" con los flujos identificados (Fuente: elaboración propia)

En la siguiente figura se muestra el grafo de flujo obtenido:

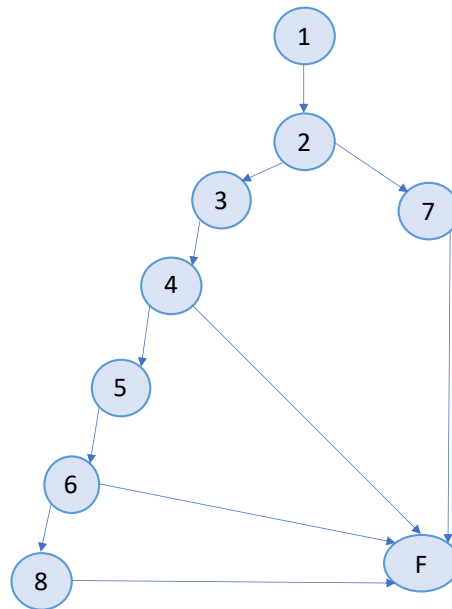


Figura 15 Grafo de flujo asociado a la implementación del método *sendUnlockConnectorRequest* (Fuente: elaboración propia)

**Paso 2:**

Cálculo de la complejidad ciclomática correspondiente al grafo precedido.

$$V(G) = R = 4$$

$$V(G) = A - N + 2 = 4$$

$$V(G) = L + 1 = 4$$

Tomando en cuenta el resultado de las tres fórmulas anteriores se determina que el método posee una complejidad ciclomática de valor 4.

**Paso 3:**

Determinar el conjunto base de rutas linealmente independientes. Dando como resultados 4 rutas linealmente independientes:

Ruta Básica 1: 1-2-7-F

Ruta Básica 2: 1-2-3-4-F

Ruta Básica 3: 1-2-3-4-5-6-F

Ruta Básica 4: 1-2-3-4-5-6-8-F

**Paso 4:**

Una vez definidas los caminos, se procede a diseñar los casos de prueba para cada una de las rutas básicas obtenidas. A continuación, se presenta los casos de pruebas definidos por las rutas básicas.

<b>Caso de prueba Camino Básico #1</b>	
<b>Descripción:</b> Este caso de prueba se ejecuta cuando se intenta enviar el mensaje al punto de carga y falla la comunicación o no se obtiene una respuesta.	
<b>Entrada</b>	uuid = "ef41a556-fdff-46e5-b5f6-a0b1d5c2141d" connectorId = "1"
<b>Resultado</b>	La operación muestra en consola el error con su descripción, y se obtiene un valor de retorno falso.
<b>Condiciones</b>	Fallo de comunicación o demora de respuesta de la solicitud.

Tabla 41 Caso de Prueba de caja blanca para el camino básico 1 (Fuente: Elaboración propia)

<b>Caso de prueba Camino Básico #2</b>	
<b>Descripción:</b> Este caso de prueba se ejecuta cuando se recibe un mensaje de solicitud incorrecto.	
<b>Entrada</b>	uuid = "ef41a556-fdff-46e5-b5f6-a0b1d5c2141d" connectorId = "1"
<b>Resultado</b>	Valor de retorno falso.
<b>Condiciones</b>	El mensaje de respuesta sea inválido.

Tabla 42 Caso de Prueba de caja blanca para el camino básico 2 (Fuente: Elaboración propia)

<b>Caso de prueba Camino Básico #3</b>	
<b>Descripción:</b> Este caso de prueba se ejecuta cuando la respuesta de la solicitud indica que no se pudo desbloquear el conector.	
<b>Entrada</b>	uuid = "ef41a556-fdff-46e5-b5f6-a0b1d5c2141d" connectorId = "1"
<b>Resultado</b>	Valor de retorno falso.
<b>Condiciones</b>	El conector no se pueda desbloquear.

Tabla 43 Caso de Prueba de caja blanca para el camino básico 3 (Fuente: Elaboración propia)

<b>Caso de prueba Camino Básico #4</b>	
<b>Descripción:</b> Este caso de prueba se ejecuta cuando la respuesta de la solicitud indica que se pudo desbloquear el conector.	
<b>Entrada</b>	uuid = "ef41a556-fdff-46e5-b5f6-a0b1d5c2141d" connectorId = "1"
<b>Resultado</b>	Valor de retorno verdadero.
<b>Condiciones</b>	El conector se pueda desbloquear.

Tabla 44 Caso de Prueba de caja blanca para el camino básico 4 (Fuente: Elaboración propia)



### CAPÍTULO III: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

Se detectaron 4 casos de prueba con la técnica de camino básico y con la ayuda de la herramienta Junit5 se realizó la ejecución de los casos de prueba.

En la siguiente figura se muestra la implementación que responde al “Caso de prueba Camino Básico #4” con el uso de la herramienta.

```
@Test
void sendUnlockConnectorRequest_Failure() throws Exception {
    // Configuración de datos de prueba
    UUID uuid = UUID.randomUUID();
    Integer connectorId = 1;

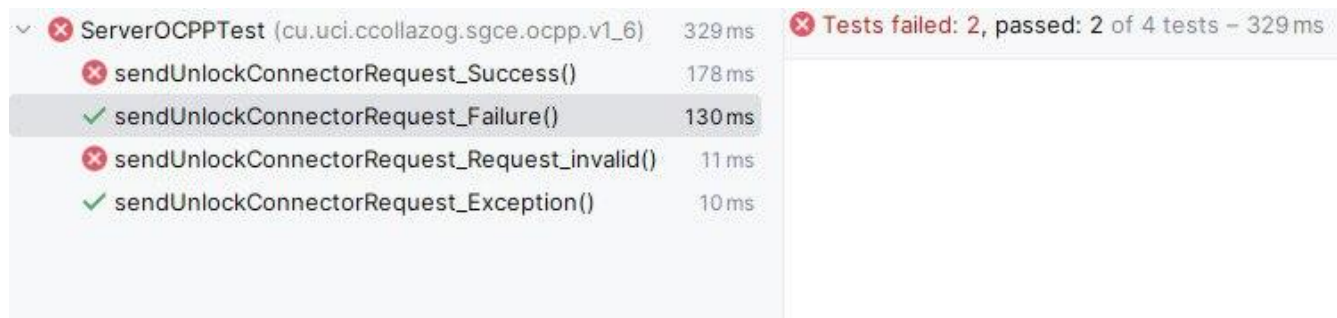
    UnlockConnectorRequest expectedRequest = new UnlockConnectorRequest(connectorId);
    UnlockConnectorConfirmation unlockConfirmation = new UnlockConnectorConfirmation(UnlockStatus.UnlockFailed);
    CompletableFuture<Confirmation> future = CompletableFuture.completedFuture(unlockConfirmation);

    // Configuración de comportamiento de los mocks
    when(serverCoreProfile.createUnlockConnectorRequest(connectorId)).thenReturn(expectedRequest);
    when(jsonServer.send(uuid, expectedRequest)).thenReturn(CompletableFuture.completedFuture(unlockConfirmation));
    when(sessionStore.getChargePointId(uuid)).thenReturn("PuntoDeCarga123");
    // Ejecución de la función a probar
    boolean result = serverOcpp.sendUnlockConnectorRequest(uuid, connectorId);

    // Verificación de resultados
    assertTrue(!result, "Se espera un desbloqueo fallido");
}
```

Figura 16 Implementación del caso de prueba #4. (Fuente: elaboración propia)

Una vez implementado los casos de pruebas restantes se procede a la ejecución de los mismos, como se evidencia en la siguiente figura.



Test Name	Duration	Status
ServerOcppTest (cu.uci.collazog.sgce.ocpp.v1_6)	329 ms	Failed
sendUnlockConnectorRequest_Success()	178 ms	Failed
sendUnlockConnectorRequest_Failure()	130 ms	Passed
sendUnlockConnectorRequest_Request_invalid()	11 ms	Failed
sendUnlockConnectorRequest_Exception()	10 ms	Passed

Tests failed: 2, passed: 2 of 4 tests - 329 ms

Figura 17 Resultados arrojados por la herramienta Junit 5 (Fuente: elaboración propia)

Como resultados de las 4 pruebas realizadas se obtuvieron 2 No Conformidades (NC). Se realizaron un conjunto de acciones correctivas al código para mitigar estas NC encontradas.

Se realiza una segunda iteración donde no se obtuvieron No Conformidades (Ver figura 18).

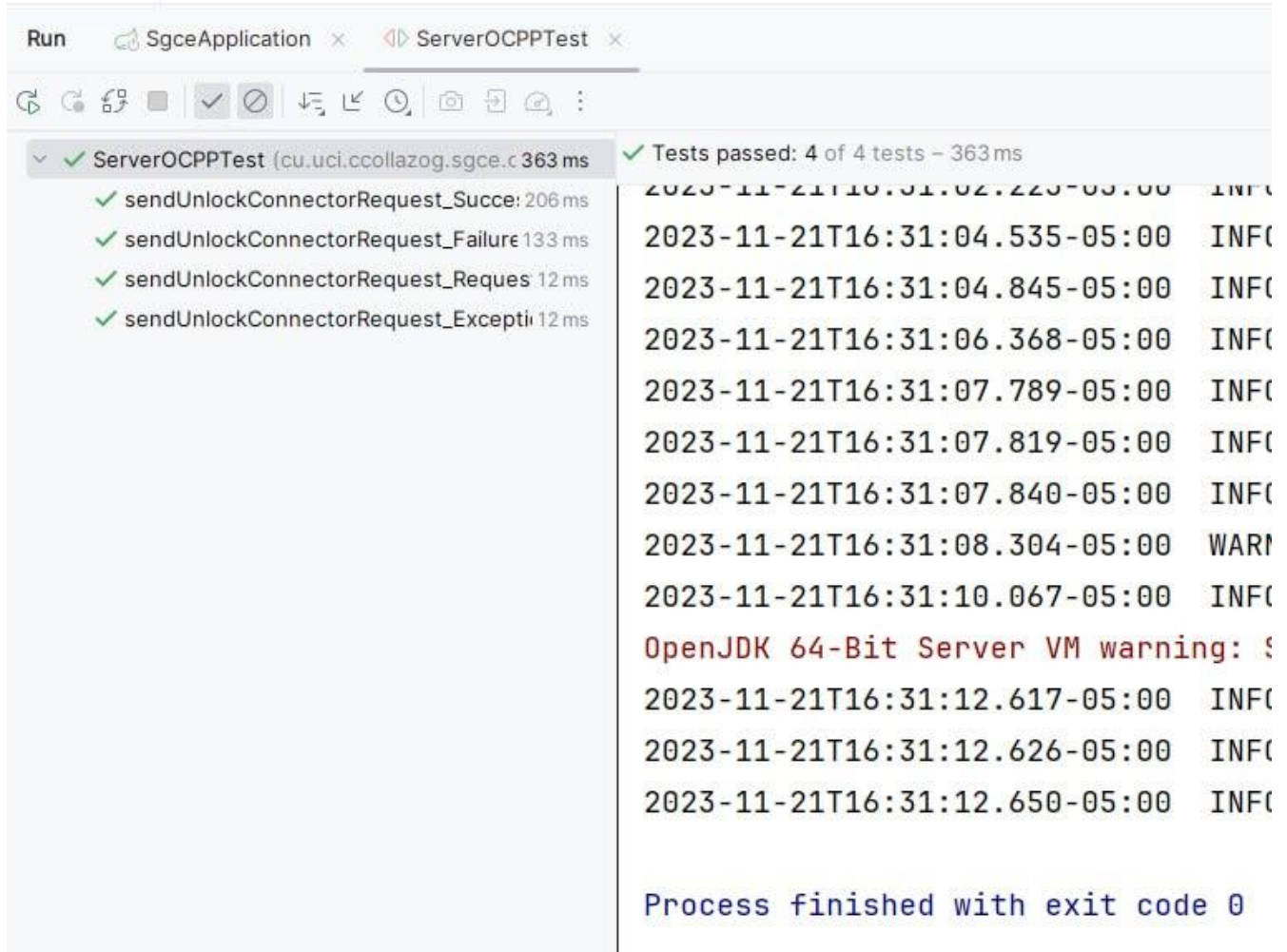
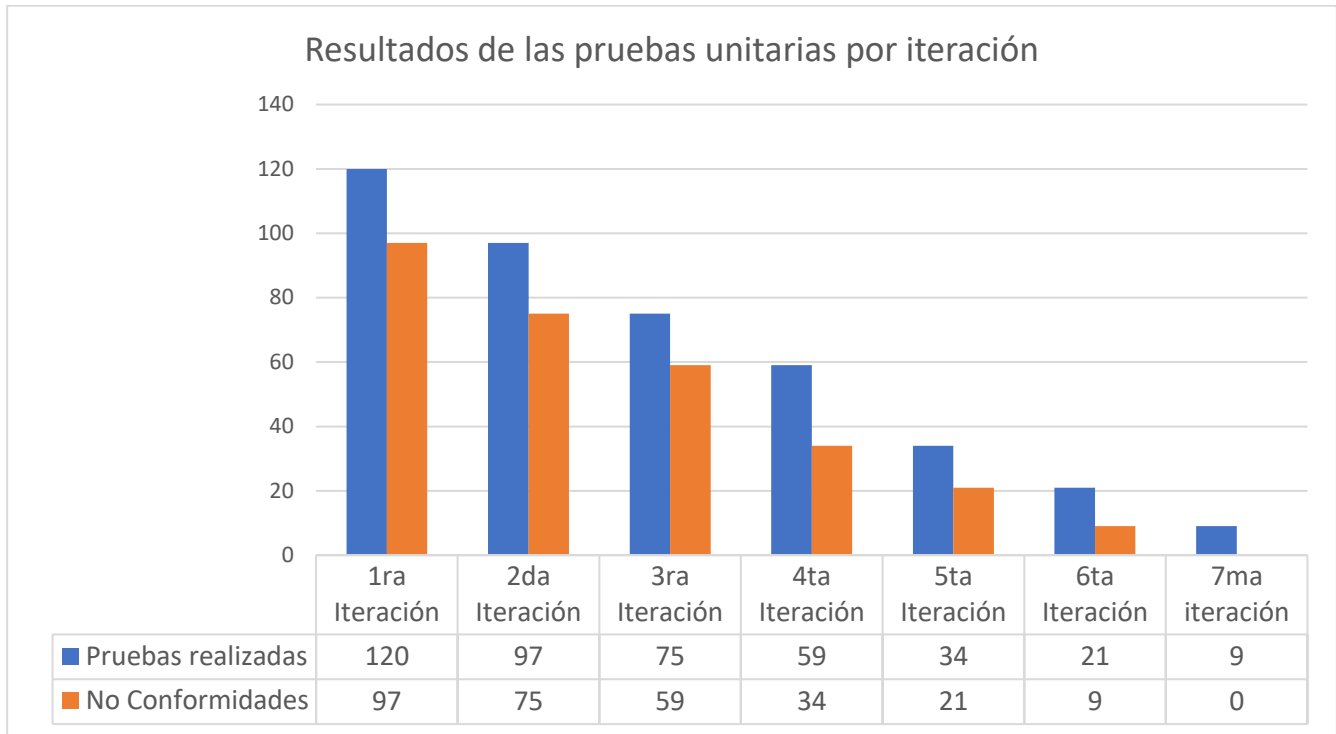


Figura 18 Resultados arrojados por la herramienta Junit 5 en la segunda iteración. (Fuente: elaboración propia)

En la figura 21 que se muestra a continuación se realiza un análisis mediante un gráfico de barras de las todas las pruebas realizadas durante este nivel de prueba.



*Figura 19 Resultados de las pruebas unitarias por iteración (Fuente: elaboración propia)*

Como se refleja en la gráfica, se identificaron un total de 97 No Conformidades en la 1ra iteración, las cuales fueron abordadas y corregidas en iteraciones posteriores. Tras completar 7 iteraciones, se logró mitigar todas las No Conformidades detectadas, cumpliendo así con el objetivo establecido para las pruebas a nivel de unidad.

### 3.2.2 Nivel de integración. Métodos de Caja Negra

Las pruebas de integración son una técnica sistemática para construir la arquitectura del software al mismo tiempo que se realizan pruebas para descubrir errores asociados con la interfaz. El objetivo es tomar componentes probados a nivel unitario y construir una estructura de programa que haya sido dictada por el diseño (Pressman & Maxim, 2015).

La caja negra especifica el comportamiento de un sistema o una parte de un sistema. El sistema (o parte) responde a estímulos específicos (eventos) aplicando un conjunto de reglas de transición que mapean el estímulo en una respuesta (Pressman & Maxim, 2015).

Para probar el funcionamiento a nivel de integración del sistema se utiliza la herramienta Postman que es una aplicación que nos permite testear servicios web.

Métodos de pruebas que se pueden realizar con Postman:

- GET: obtener información.

- POST: agregar información.
- PUT: reemplazar la información.
- PATCH: actualizar alguna información.
- DELETE: borrar información.

A continuación, se detallan las pruebas de integración realizadas correspondientes a los requisitos funcionales “Registrar Token”, “Obtener lista de tokens”, “Eliminar Token”.

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	none	1	1s 994ms	14	119 ms

All Tests Passed (10) Failed (4) Skipped (0)

Iteration 1

**POST** Crear Token  
http://localhost:8080/tokens

- FAIL Estado de respuesta: 200 | AssertionError: expected response to have status code 201 but got 200
- FAIL La respuesta tiene los valores requeridos | AssertionError: expected 'PHASE\_04' to be NaN
- PASS Validar que el campo 'uid' no es una cadena vacía
- PASS Validar que el campo 'type' no es una cadena vacía
- PASS Validar que el campo 'group\_id' no es una cadena vacía
- PASS Validar que el campo 'last\_updated' no es una cadena vacía
- PASS El campo 'last\_updated' es un objeto de fecha

**GET** Lista de tokens  
http://localhost:8080/tokens

- PASS Response status code is 200
- FAIL Response is an array with at least one element | AssertionError: expected [ { uid: 'PHASE\_01', ...(4) }, ...(1) ] to have a length at least 20 but got 2
- FAIL The uid field is a non-empty string | AssertionError: Value should not be empty: expected 'PHASE\_01' to have a length at least 10 but got 8
- PASS Valid field is a boolean value

**DELETE** Eliminar token  
http://localhost:8080/tokens/PHASE\_04

An error occurred while running test scripts for this request. [Check Console](#)

Figura 20 Resultados arrojados por la herramienta Postman (Fuente: elaboración propia)

Como resultado de los 3 requisitos probados se obtuvieron 3 No Conformidades (NC). Se realizaron un conjunto de acciones correctivas al código para mitigar estas NC encontradas.

Se realiza una segunda iteración donde no se obtuvieron No Conformidades (Ver figura 21).

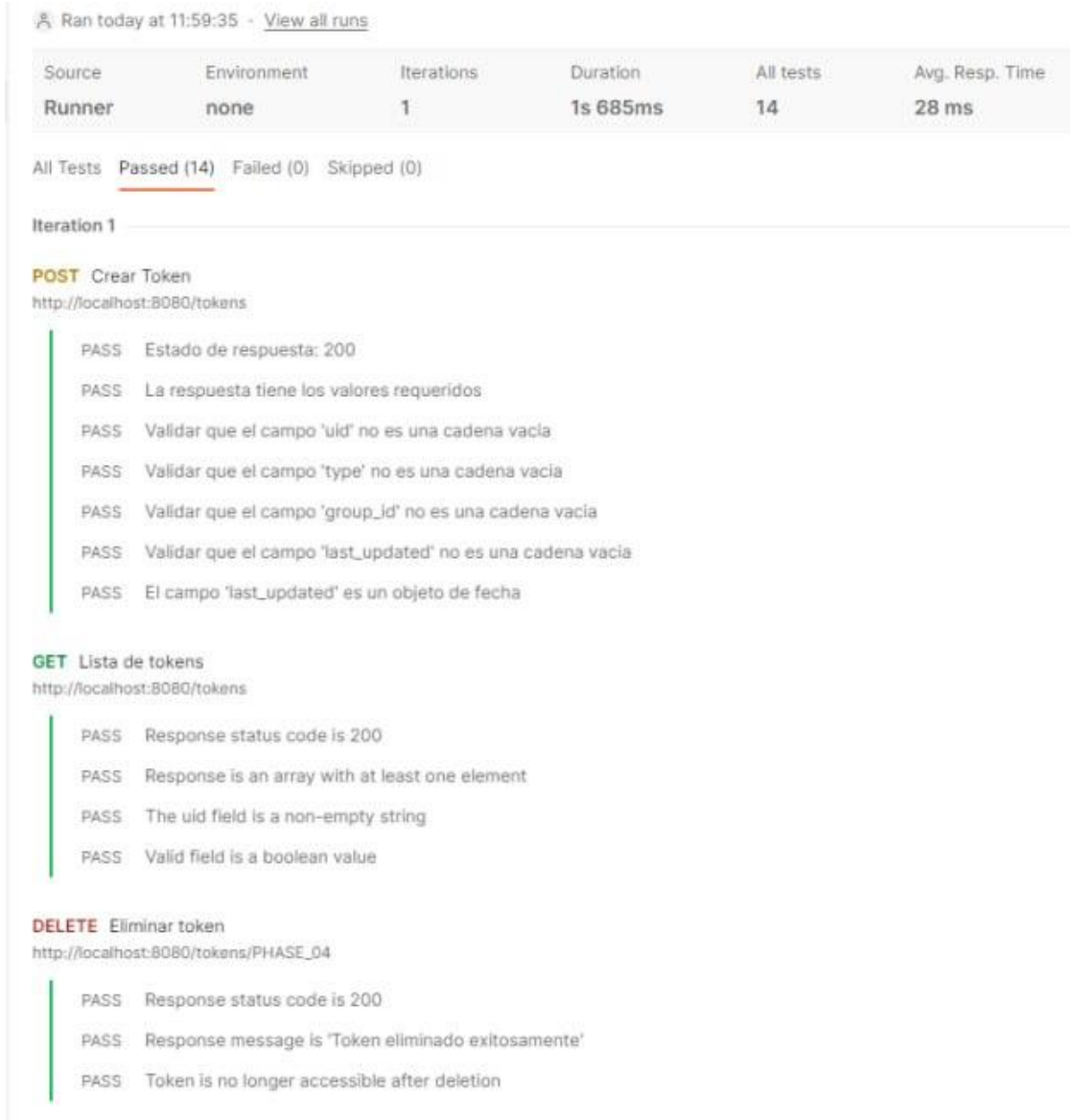


Figura 21 Resultado arrojados por la herramienta Postman en la segunda iteración. (Fuente: elaboración propia)

En la figura 22 que se muestra a continuación se realiza un análisis mediante un gráfico de barras de las todas las pruebas realizadas durante este nivel de prueba.

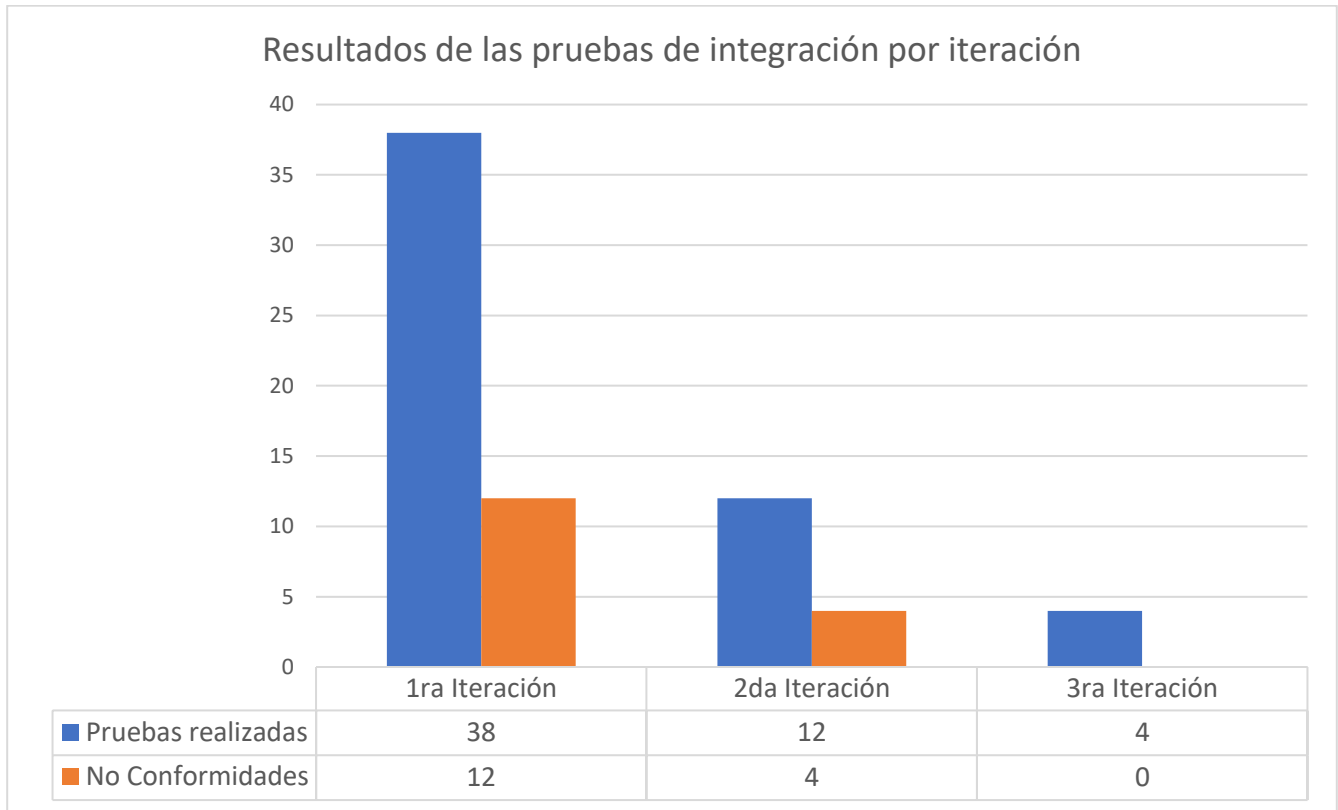


Figura 22 Resultados de las pruebas de integración por iteración (Fuente: elaboración propia)

Una vez ejecutadas y obtenido todos los resultados de las pruebas realizadas para todos los requisitos a través de la herramienta Postman, se obtuvo en una primera iteración un total de 12 No Conformidades. Tras completar 4 iteraciones, se logró mitigar todas las No Conformidades detectadas, cumpliendo así con el objetivo establecido para las pruebas a nivel de integración.

### Conclusiones del capítulo

- Mediante la realización de las pruebas a nivel de unidad se encontraron un total de 97 No Conformidades. Las cuales pudieron ser corregidas en 7 iteraciones.
- Con la ejecución de las pruebas a nivel de integración se obtuvieron 12 No Conformidades. Las cuales pudieron ser corregidas en 3 iteraciones garantizando la integridad del sistema en su conjunto.
- La ejecución de pruebas funcionales evidencia que las funciones tanto externas como internas son operativas, que las entradas se aceptan de forma adecuada y que se producen salidas correctas, obteniendo finalmente una aplicación que satisface los requisitos definidos.

## CONCLUSIONES FINALES

Con el desarrollo del servicio de gestión de puntos de carga se puede afirmar que:

- La sistematización del marco teórico de la investigación sustentó los principales referentes teóricos en los que se enmarca el desarrollo de la propuesta de solución.
- La identificación de los requisitos, así como el análisis y diseño del servicio web, permitieron obtener una aproximación y sentar las bases para la implementación de dicho servicio.
- El desarrollo del servicio web permitió obtener un sistema funcional que contribuye a un control sobre las estaciones de carga.
- El empleo de las pruebas de caja blanca y caja negra desempeñaron un papel crucial en la detección temprana de errores, la mejora de la calidad del software y la validación de requisitos.
- Se obtuvo el documento de tesis como memoria escrita.

## RECOMENDACIONES

Se recomienda implementar los demás perfiles definidos en el protocolo OCPP para extender las funcionalidades de gestión en las estaciones de carga.



**REFERENCIAS BIBLIOGRÁFICAS**

- adminetecnic. (2019, enero 14). Mobility Smart Manager: Software de gestión para puntos de recarga Etecnic - Oferim servei 360 de punts de recàrrega per a vehicles elèctrics. Etecnic - Oferim servei 360 de punts de recàrrega per a vehicles elèctrics. <https://etecnic.es/noticias/vive-etecnic/swetecnic/mobility-smart-manager-software-de-gestion-para-puntos-de-recarga/>
- Ampcontrol Technologies. (2023). EV Charging Management Software—Ampcontrol. <https://www.ampcontrol.io/smart-charging-software>
- Asana. (2022, enero 21). Historias de usuario: 3 ejemplos para generar valor para el usuario [2022] • Asana. Asana. <https://asana.com/es/resources/user-stories>
- Betania V. (2023, mayo 3). ¿Qué es un sistema de gestión de bases de datos? Explicación de SGBD. <https://www.hostinger.es/tutoriales/sghbd>
- ChargeLab Inc. (2023). EV charger management software—ChargeLab. <https://charge-lab.co/software>
- Daynelis Brito Morales, Johan Bravo Borrell, & Lijandy Jiménez Armas. (2019). Aplicación móvil para el análisis de la información captada en SIGEv3.0. 12(6), 55-71.
- Figueredo, L. (2021). Proceso de pruebas de software para un modelo de calidad en Cuba. I+D Tecnológico, 17(1), Article 1. <https://doi.org/10.33412/idt.v17.1.2914>
- Global EV – Analysis. (2021). IEA. <https://www.iea.org/reports/global-ev-outlook-2021>
- greenlots. (2018). OCA - Open Standards Whitepaper. <https://www.openchargealliance.org/uploads/files/OCA-Open-Standards-White-Paper-compressed.pdf>
- IBM Corporation. (2021, marzo 5). Class Diagram. <https://www.ibm.com/docs/en/rsm/7.5.0?topic=structure-class-diagrams>

- Idea IntelliJ—Introducción. (s. f.). Recuperado 26 de mayo de 2023, de [https://www.tutorialspoint.com/intellij\\_idea/intellij\\_idea\\_introduction.htm](https://www.tutorialspoint.com/intellij_idea/intellij_idea_introduction.htm)
- jetbrains. (2023). Features—IntelliJ IDEA. JetBrains. <https://www.jetbrains.com/idea/features/>
- JUnit. (2021, febrero 13). JUnit – Acerca de. <https://junit.org/junit4/>
- Khoirom, M. S., Sonia, M., Laikhuram, B., Laishram, J., & Singh, T. D. (2020). Comparative Analysis of Python and Java for Beginners. 07(08).
- Lesjak, Ž. (2023, octubre 2). Protocolos y normas de recarga de vehículos eléctricos: Una guía completa. Tridens. <https://tridenttechnology.com/es/normas-de-los-protocolos-de-recarga-de-vehiculos-electricos/>
- Metodologías de desarrollo de software. (2022, enero 20). Domain Logic. <https://domainlogic.io/metodologias-de-desarrollo-de-software-2022/>
- microsoft. (2023). ¿Qué es el modelado de datos? | Microsoft Power BI. <https://powerbi.microsoft.com/es-es/what-is-data-modeling/>
- Open Charge Alliance. (2017). Open Charge Point Protocol 1.6 edition 2.
- O'Reilly Media. (2023). 1. Arquitectura en capas: Patrones de arquitectura de software [Libro]. <https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html>
- Phoenix Contact. (2023). Software para la gestión de cargas | Phoenix Contact. <https://www.phoenixcontact.com/es-pc/productos/tecnologia-de-carga-para-la-electromovilidad/gestion-de-carga-y-recarga#ex-content-transclusion-snippet--266>
- Ponce Briones, D. K. (2016). Análisis Comparativo De Los Entornos De Desarrollo Integrados (Ide): Eclipse, Netbeans Y Jdeveloper Para El Desarrollo De Aplicaciones Java Enterprise Edition [Thesis, Universidad de Guayaquil. Facultad de Ciencias Matemáticas y

- Físicas. Carrera de Ingeniería en Sistemas Computacionales]. <http://repositorio.ug.edu.ec/handle/redug/15862>
- Pressman, R. S., & Maxim, B. R. (2015). *Software engineering: A practitioner's approach* (8. ed). McGraw-Hill Education.
- Priyasta, D., Hadiyanto, H., & Septiawan, R. (2023). Ensuring Compliance and Reliability in EV Charging Station Management Systems: A Novel Testing Tool for OCPP 1.6 Messages Conformance. *Journal Européen des Systèmes Automatisés*, 56, 121-129. <https://doi.org/10.18280/jesa.560116>
- Ramírez Pérez, S. (2020). Estudio del framework Spring, Spring Boot y microservicios. <https://ebuah.uah.es/dspace/handle/10017/45107>
- Refactoring.Guru. (2014, 2023). ¿Qué es un patrón de diseño? <https://refactoring.guru/es/design-patterns/what-is-pattern>
- Richards, M., & Ford, N. (2020). *Fundamentals of software architecture: An engineering approach* (First edition). O'Reilly Media, Inc.
- SanghpriyaGautam. (2023, marzo 29). Introduction to Programming Languages. Geeksfor-Geeks. <https://www.geeksforgeeks.org/introduction-to-programming-languages/>
- Sommerville, I. (2016). *Software engineering* (Tenth edition). Pearson.
- Sparx Systems. (2023). Sparx Systems—Tutorial UML 2—Diagrama de Despliegue. [http://www.sparxsystems.com.ar/resources/tutorial/uml2\\_deploymentdiagram.php](http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.php)
- tecnologias-informacion. (2018). Modelado de Datos: Definición, Usos y Tipos. <https://www.tecnologias-informacion.com/modeladodatos.html>
- The PostgreSQL Global Development Group. (2023, abril 5). PostgreSQL: About. <https://www.postgresql.org/about/>

- van Amstel, M., Ghatikar, R., & Wargers, A. (2021). Importance of open charge point protocol for the electric vehicle industry. Open Charge Alliance, ND, Accessed on November, 10. [https://www.openchargealliance.org/uploads/files/OCA-EN\\_whitepaper\\_OCPCP\\_vs\\_proprietary\\_protocols\\_v1.0.pdf](https://www.openchargealliance.org/uploads/files/OCA-EN_whitepaper_OCPCP_vs_proprietary_protocols_v1.0.pdf)
- Visual Paradigm. (2022). ¿Qué es el lenguaje de modelado unificado (UML)? <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/>
- Visual Paradigm. (2023). Descripción general del producto Visual Paradigm. [https://www.visual-paradigm.com/support/documents/vpuserguide/12/13/5963\\_visualparadi.html](https://www.visual-paradigm.com/support/documents/vpuserguide/12/13/5963_visualparadi.html)
- Zorko, L. (2023, enero 8). Sistema de gestión de estaciones de recarga para vehículos eléctricos—Supervisión y gestión. Tridens. <https://tridenttechnology.com/es/sistema-de-gestion-de-estaciones-de-carga-ve/>

## ANEXOS

## ANEXO 1: Carta de aceptación



## ACTA DE ACEPTACIÓN

### ACTA DE ACEPTACIÓN

En cumplimiento del Convenio con el Centro de Tecnologías Interactivas (VERTEX) y en función del proyecto: **Plataforma de Servicios de electro movilidad para el ecosistema de carga pública de Cuba**. Se hace entrega del producto que se relaciona a continuación:

- **Servicios Web para la gestión de las estaciones de cargas de vehículos**

La Parte Cliente , luego de haber revisado los productos de trabajo, determina que se Aceptan . Se determina que los requisitos pactados fueron satisfechos y que la calidad del artefacto final es la adecuada;

Entrega: Tesista de 5to años	Recibe: Centro de Tecnologías Interactivas
Nombre y Apellidos: Claudia Collazo Gumà	Nombre y Apellidos: Andy Suárez Oña Cargo: Jefe de Proyecto
Firma 	Firma 



Fecha: 15/11/2023